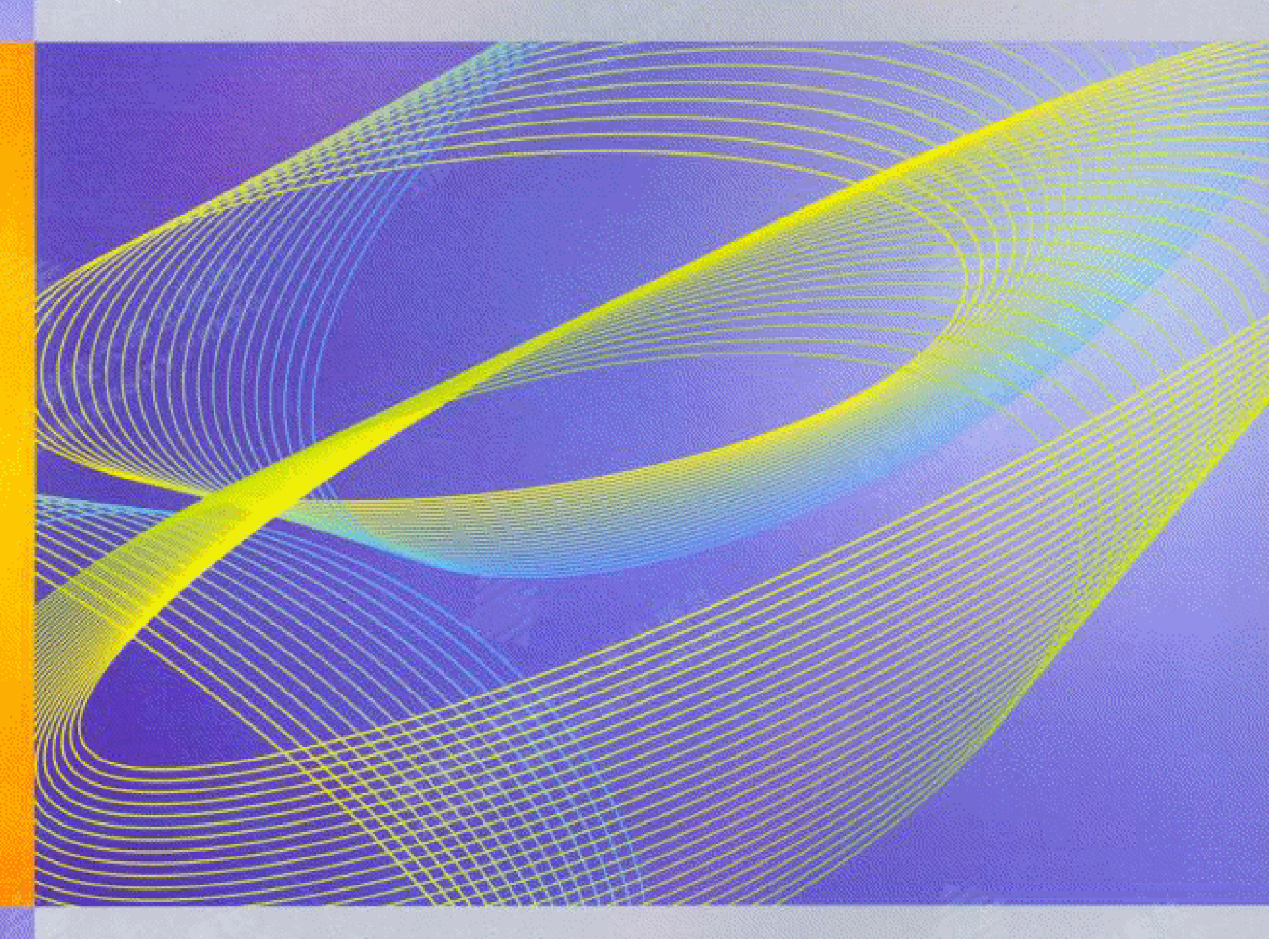
MATLAB

（第 二 版）

MATLAB 及其在理工课程中的应用指南



□ 陈怀琛 编著

西安电子科技大学出版社
http:// www.xduph. com

MATLAB 及其在理工课程中的应用指南

（第 二 版）

陈怀琛　编著

西安电子科技大学出版社
2004

内 容 简 介

本书由语言篇和应用篇两部分组成。语言篇介绍MATLAB语言的基本语法，既便于自学，又有计算机光盘配合教学，适合于作为集体教学的教材；应用篇给出用MATLAB语言解题的80多个实例，涉及的课程范围主要有高等数学、大学物理、力学、机械、电工电子和信号系统等。这些例题使用了MATLAB中多方面的语句，有助于提高编程的技巧，通过其中的程序可以大大地提高各课作业的效率。书中全部程序可以从网上免费下载。本书是21世纪理工科大学生提高学习效率的必备工具书。

本书的适用范围：一是作为大学生学习MATLAB语言的入门教材；二是作为学生在大学期间做习题的参考书；三是供各课的教师作为讲课、演示和解题的工具；四是作为工程技术人员自学MATLAB的手册。

图书在版编目(CIP)数据

MATLAB及其在理工课程中的应用指南/陈怀琛编著.

2版—西安：西安电子科技大学出版社，2004.11

ISBN 7-5606-0781-0

Ⅰ. M...　Ⅱ. 陈...　Ⅲ. 计算机辅助计算—软件包，MATLAB　Ⅳ. TP391.75

中国版本图书馆**CIP**数据核字(**2004**)第**098569**号

策　　划　毛红兵

责任编辑　潘恩祥　毛红兵

出版发行　西安电子科技大学出版社(西安市太白南路2号)

电　　话　(029)88242885　88201467　　邮编　710071

http://www.xduph.com　　E-mail：xdupfxb@pub.xaonline.com

经　　销　新华书店

印刷单位　陕西华沐印刷科技有限责任公司

版　　次　2000年1月第1版　2004年11月第2版　2004年11月第4次印刷

开　　本　787毫米×1092毫米　1/16　印张　15.5

字　　数　357千字

印　　数　14 001～18 000册

定　　价　20.00元

ISBN 7-5606-0781-0/TP·0402

XDUP　1052012-4



序

半个世纪以来，信息科技特别是计算机技术的飞速发展，大大加速了社会的改革进程。利用计算机不仅能使人们摆脱繁重的体力劳动，更快捷、更精确地进行生产，而且借助于计算机辅助设计(CAD)和辅助制造(CAM)，乃至计算机集成制造系统(CIMS)，可使企业的生产效率大幅度提高。

信息科技发展对高等教育的影响是深远的，特别是在理工科教学方面，普遍增设了计算机类的课程，使学生能够适应将来的工作环境。其实，在大学教育里，利用计算机手段提高教学效率，并使学生在实用中掌握计算技能同样是十分重要的。现在的中、老年教师都会记得曾经使用计算尺和电子计算器，用来做一般的算术和简单的函数运算。计算机，特别是微机的出现和普及，使原来因计算复杂而难以实现的问题得到了解决，有可能在教学中不再回避复杂计算，而将问题的分析引向更深的层次。

计算机的应用离不开计算语言，FORTRAN、BASIC ……已成功地应用于各种场合，但作为科学和工程问题，更多的是在分析计算(如常用的矩阵计算和复杂的函数运算)和形象地图示等方面，应用通常的计算语言并不方便。为此，在20世纪80年代初期，推出了多种科学计算语言。MATLAB就是应用最广泛的语言之一。它的特点是与科技人员的思维方式和书写习惯相适应，操作简易，人机交互性能好，从而使广大科技人员乐于接受。

基于以上原因，国外有许多理工科的书籍和教材已将MATLAB作为专用的科学计算语言融入专业内容之中，并从大学一年级就开始使用这种语言。实践表明，特别是对一些数值计算广泛应用的专业，教学效率和效果的提高是非常明显的。

过去，在MATLAB计算语言的使用上，国内高校与国外高校相比有较大的差距，客观原因是硬件条件较差，许多高校还不能为低年级学生提供必要的设备。近年来，情况已经有了很大的变化：不仅学校的设备条件得到了改善，而且许多学生都有了自己的微机。这就使理工科学生完全有可能将MATLAB这一科学计算语言学好用好，使之成为自己熟练掌握的工具，这会对自己提高当前学习效率和今后的工作带来较大裨益。

陈怀琛教授热心祖国教育事业，他在美国访问期间做了广泛的调查，并为西安电子科技大学购买了MATLAB的教学版。为了从大学一年级开始就能在许多课程里应用它，陈教授又与众多的基础课和专业基础课教师进行了多次探讨，并在学校开办了讲习班，收到了良好的效果。

为了能将这一工作在国内更快地推广，他又编写了这本应用指南。我认为将MATLAB用于各个理工科课程是一件刻不容缓的事，本书的出版将对这项工作起到推动作用。

保　铮　谨识

1999年11月

于西安电子科技大学

第二版前言

本书的第一版出版于 2000 年 1 月，至今已近五年，仍需求不断。计算机语言的书籍通常的生命周期是不长的，而这本书却能延续四年以上，其原因可能有三个方面：

首先因为在这近五年中，MATLAB 在全世界的工程界应用日益广泛，国内的高校和科研部门也普遍使用。在大学各门课程中使用 MATLAB 进行教学的潮流正迅速发展。愈来愈多的学校都认识到，对理工科的大学生，MATLAB 就是替代计算尺和计算器的最好工具，应当在大学阶段学会使用。学得愈早，受益就愈多。许多大学在一年级下学期就开始安排“MATLAB 入门”课程，有些大学则安排在二年级。这就提出了对更多的 MATLAB 教材的需求，本书的编写思想恰好适应了这个潮流。

第二是本书有自己的学科特色，那就是把重点放在 MATLAB 语言在理工课程的应用方面。目前关于 MATLAB 的书籍不下几十种，入门教材大多数只联系数学课，本书则还联系了物理、机械、电机、信号和系统等广泛的领域。书的难度比只联系数学要大，但培养了读者用计算机来解决各种课程问题的独立的工作能力。学会从问题提出、建模、用 MATLAB 编程、到结果的显示讨论的全过程。我们认为，培养学生的这个能力十分重要，可使他们终身受益。纵观各种 MATLAB 书籍，具备这个特色的并不多。但这也引出了本书的一个缺点，全书涉及课程较多，学生虽然都要学，但老师不好当。解决的方法是：教师只讲一部分，包括语言入门和教师熟悉的学科中的应用，其余由学生自学，要求他们写出解决特定或自选题目的报告。

第三则是因为许多 MATLAB 书籍实际上是手册的译本，主要作为英文不过关的读者学习手册之用，版本一升级，手册也过时了。本书则完全是自编的，其语言篇只讲主要语法，尽量的与版本无关。而应用篇中的题目都是从各门课程中引出的，所有的程序全是独立编成的。所以，读了手册或别的 MATLAB 书，再读本书中的应用篇，仍然感到是新的，不会感到重复。尽管语言不断升级，用它来解题的思路和应用程序并不会改变。其实本书中的解题建模方法，可以用于解决一些高深的问题，所以我们的读者群中还有高年级学生和研究生。

在修订中，我们继续保持了这些长处，应用篇基本上没有改动。

第二版所做的主要变动如下：

(1) 在这四年中，作者还写了两本书，《MATLAB 及在电子信息课程中的应用》和《数字信号处理教程——MATLAB 释义与实现》。深感计算工具的改进可以反过来对理论建模产生重大影响，很想在本书中反映这个思想。不过在简单问题中不好体现，所以在本书中，专门把例 7-3-3 的高阶振动问题改了一下，以说明这个观点。对机械振动的三个例题，在第一版中用的都是传统的建模方法，只改用 MATLAB 做计算工具，所以仍然只能解简单的二阶问题。在第二版中把例 7-3-3 中的建模方法改用为矩阵，以适应 MATLAB 工具，就可以很简捷地解高阶的复杂题目。

(2) 第一版出版于2000年1月，当时是以MATLAB 4.2和5.1版本为基础写的。2004年5月MATLAB已升级到了7.0版本，尽管它的语法并没有太大变化，但使用界面还是大不相同的需要进行修订。对初学者而言，用最高最新的界面未必有利。而MATLAB 6.x已经流行了三年半，是目前最普及的版本，而且7.0版本的界面改变不大。因此，“语言篇”全部按照6.x的界面进行了修改。

(3) 为语言篇增加了一些MATLAB习题，以便于读者自学。

(4) 从2003年7月起，我们开始免费赠送本书的全部例题程序，在网址 http://www.xduph.com 下搜索本书名，即可免费下载。如果下载不成功，读者可以给作者发电子邮件索取，邮箱地址为 hchchen@xidian.edu.cn，请提供单位和姓名，我们将在回函的附件中提供。

(5) 关于本书的语言篇，我们原来制作了四小时的录像带，后改为四张VCD，主要提供课堂教学之用，现在作者准备以MATLAB 6.x和7.0为背景，重新制作一套在计算机上放映的光盘，使其在内容和质量上都有所提高。有关光盘的购买，可与作者或西电网络学院办公室联系。有关本书的改进意见，读者可通过电子邮件或打电话(029)88202988向作者提出，本人热忱欢迎。

陈怀琛

2004年9月5日

于西安电子科技大学

第一版前言

1. 为什么要写这本书？

从20世纪80年代起，出现了科学计算语言，也称为数学软件。因其高效、可视化和推理能力等特点，在大学教育和科学研究中，正迅速取代FORTRAN和BASIC语言。这类语言中已商品化的有MATLAB、MATHEMATICA、MATHCAD、MAPLE等，它们的功能大同小异，又各有所长。目前在工程界流行最广的是MATLAB语言，这种语言首先在研究生课程中应用，如自动控制和信号处理等课程，并开始有这方面的教材，随后在各种课程中广泛使用。根据最近因特网上的检索，美国已有300多种有关MATLAB语言的书籍，仅Prentice-Hall出版社近3年内出版的将MATLAB用于各门课程的教材就超过百种，其范围包括：微积分、矩阵代数、应用数学、物理、力学、信号与系统、电子线路、电机学、机械振动、科学计算、有限元法、计算机图形学、自动控制和通信技术等。

这种算法语言为何能大大提高教学的效率呢？

(1) 它可用一种几乎像通常笔算式的简练程序，把繁琐的计算交给计算机去完成。

(2) 由于它的表达式简练而准确，往往可以简化公式的推导和概念的叙述。

(3) 它可以方便迅速地用三维图形、图像、声音、动画等表述计算结果，帮助逻辑思维。

(4) 它可很方便地把复杂的计算过程凝聚成一个程序，以后可随意调用，避免教学中的重复。

(5) 它的可扩展性强，在学好其基础部分之后，还有几十种工具箱可用于各类科研需要，这可缩短学习和实践工作的距离。

由于这些特点，我认为，应该把MATLAB作为一种贯穿大学学习全过程的语言教给学生。这就是说，① 应该使一年级大学生就初步学会这种语言；② 应该在以后的各门主要课程中不断地反复应用和深化。

近几年来，有关MATLAB语言的书籍在我国逐渐增多，已有十多种，但它们都不适用于低年级本科教学。为了使各科的老师看到MATLAB在相关课程中的应用价值，为了指导学生在学习各门课程中能利用MATLAB语言解题，我们编写了这本教材。

2. 本书的构成

本书包括语言篇和应用篇两篇。

第一篇为语言篇。介绍MATLAB语言的基础。这部分内容既可自学，也可与西安电子科技大学电教中心出版的录像带配套使用，该录像带共有4节课(每节课50分钟)，以一年级大学生为对象。在MATLAB的基础部分中，那些大学本科用不到的内容，我们只作简述并用小字印刷。本书不使用MATLAB的工具箱，一是因为大学三年级以前用不到，二是过早应用工具箱不利于低年级学生理解概念和掌握编程。

第二篇是应用篇。它是MATLAB语言在大学课程中的应用举例，其中列举了大学本

科(以电子和机械专业为主)的十多门基础课程中使用 MATLAB 语言的近百个示例。这些例题能启发学生应用的兴趣,并提高他们的编程技巧。实际上,由于 MATLAB 语言与数学基础有密切关系,学生不可能在学习语言入门后就马上掌握各种应用。通过应用篇,大学生可随着知识的增长,从一年级到三年级一直把这本书用作参考书。三年级以后的有些课程需要 MATLAB 语言的控制系统工具箱或信号处理工具箱,读者还需阅读专门的书籍。

为了使本书能作为一本指南和手册,本书中列出了 MATLAB 的全部基本函数,并采用了多种索引方法。对一些重要的函数给出了它们的应用例题,以便查阅它们的用法,并列出了按字母排序的 MATLAB 函数索引,以便读者阅读程序时反向查找。在每个例题中也指出了其语法和编程的特点。

3. 在本科教育中使用 MATLAB 语言对提高教学的效率十分有益

人类的知识正以指数规律飞速增长,21 世纪将是知识经济的时代。使我们年轻的一代,以最高的效率掌握人类已有知识的精华,又能以最快的速度和现代化方法去创新和探索,这是我们高等教育界的奋斗目标。

我们知道,借助于计算机辅助设计和制造(CAD 和 CAM),设计业和制造业已大大地提高了效率,创造了空前的物资财富。在教学领域,如果能像设计业和制造业那样利用计算机,把师生从繁琐重复的低级劳动中解放出来,把更多的时间用于概念的思考,那么教学的效率也必然大大提高。现在各大学开设某些计算机课程,只是为了学生就业的需要,很少对学生在校学习有直接的帮助。目前大学生的学习工具还是“计算器水平”,MATLAB 语言在大学教学中的普遍推广,可以与设计业中广泛应用的 CAD 相比美,它可使计算机真正成为教学的有力工具。

作者从 1995 年初开始接触 MATLAB,先是用于自动控制课程,而后用于信号处理,并且一直致力于把它推广应用于大学教学的全过程。经验说明,后者是一件很艰难的工作,需要有各课程大批教师的参与,更需要领导的大力支持,例如购买教学版软件,并创造上机条件等。本书涉及如此多的课程,也足以说明,推广 MATLAB 语言是一个有全局意义的问题,教育部门的领导应像设计和工业部门抓“甩图板”那样来抓好这件事。

4. 致谢

作者虽然已任教 46 年,教过十多门课程,但因为这本书涉及的学科领域广泛,还没有这样的书籍作为先例,写起来有相当难度。包括构思、选材、编程和注释都要从头做起,并要使程序简短易读,能被大学生看懂。在此作者对陈开周,祝向荣,刘三阳,冯晓慧,陈怀琳(北京大学),徐雄(Ohio State University),过巳吉,葛德彪,吴振森,郭立新,王德满,曾余庚,贾建援,黄一红,仇原鹰,张永瑞,冯宗哲,孙肖子,沈耀忠,戴树荪,路宏敏等(以章次排列)各位老师致谢,他们为本书提供了许多例题或程序,并提出了一些宝贵的意见。对本书的编写有很大的帮助。作者还要感谢责任编辑毛红兵,她对本书的及时出版也作出了贡献。作者还要特别感谢中科院院士保铮教授对本书的支持。

陈怀琛

1999 年 8 月 31 日

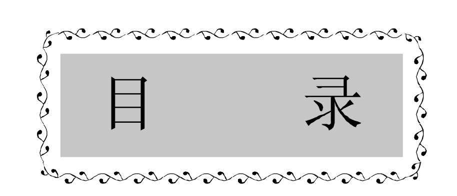
符号及标注说明

（1）由于本书涉及到大量的计算机程序，而程序中无法输入斜体和希文字母，因此为统一起见，本书中使用的符号均为正体；程序中采用国际上惯用的像形符号，例如在叙述中使用的符号 ω(希)，在程序中用 w(或 W)代替；叙述中使用的带上下标符号，如 a_1，ω_s，T_s 等，在程序中用 a1，ws，Ts 等代替。

（2）为了使全书公式与程序相统一，本书中涉及到的矢量和矩阵没有用黑体表示。

（3）在本书的图中，凡是计算机自动生成的 Y 坐标标注，字体旋转 90°，而人工生成的 Y 坐标标注，字体未旋转 90°。

（4）在应用篇中，由于各例题来自不同的领域及课程，因此程序中的符号大小写未要求统一。



目　录

第一篇　语　言　篇

第二篇　应　用　篇

第一篇　语　言　篇

本篇内容适合大学一年级下学期或二年级上学期使用。这时学生已有了一定的计算机操作技能，并且有矩阵运算的知识。这样，学生在学习本书的第1、2、3章时将不会有很大的困难。我们制作的光盘主要就针对这个部分。没有光盘的读者，只要有本书的程序集，也可以在计算机上复现书中的所有画面，很容易对照自学。

MATLAB是一种与数学水平密切相关的算法语言，第4章中介绍的内容需要较多的高等数学知识，要随着年级的增加才能逐渐深入掌握这些内容。在光盘中这部分只占20分钟，读者可根据自己的数学水平进行自学，并可与应用篇联系起来深入体会。

MATLAB中还有一些在大学本科的学习中通常用不到的内容，但在毕业设计或今后的科研工程中可能有用。为了使本书具备手册的功能，我们用小字来叙述，同时，本书用小字列出了MATLAB基本部分的全部函数库，并配以索引，便于读者查找。这部分内容可以先跳过去，待需要时再看。

第 1 章 MATLAB 语言概述

1.1 MATLAB 语言的发展沿革

MATLAB 是一种科学计算软件，主要适用于矩阵运算及控制和信息处理领域的分析设计，它使用方便，输入简捷，运算高效，内容丰富，并且很容易由用户自行扩展。MATLAB 当前已成为美国和其他发达国家在大学教学和科学研究中最常用而必不可少的工具。

MATLAB 是由美国 Mathwork 公司于 1984 年正式推出的，到 1988 年有了 3.1(Dos)版本；1992 年出了 4.1(Windows)版本；1997 年推出了 5.1(Windows)版本。2001 年初，推出了 6.1(R12)版本；2004 年夏又推出了他们的最新产品 MATLAB 7.0(R14)的正式版。一方面，随着版本的升级，其内容在不断扩充，功能也更加强大。特别是在系统仿真和实时运行等方面，有很多新进展，更扩大了它的应用前景。另一方面，版本的升级对使用环境也提出了更高的要求。不过对于学习语法基础的读者来说，各版本的差别不太大，可以从较低的版本起步。本书的全部程序是在 MATLAB 6.x 版本上通过的，但在 5.x 及 4.2c 的环境下也都能运行，只有个别语句可能要改一下。根据向上兼容的原则，在 7.0 版本下本书的全部程序当然都能运行。

MATLAB 是“矩阵实验室(Matrix Laboratory)”的缩写，它是一种以矩阵运算为基础的交互式程序语言，专门针对科学和工程计算和绘图的需求而开发的。与其他计算机语言相比，其特点是简洁和智能化，适应科技专业人员的思维方式和书写习惯，使得编程和调试效率大大提高。它用解释方式工作，键入程序立即得出结果，人机交互性能好，为科技人员乐于接受。特别是它可适应多种平台，并且随计算机硬、软件的更新及时升级。MATLAB 语言在国外的大学工学院中，特别是在数值计算用得最频繁的电子信息类学科中，已成为每个学生都掌握的工具了。它大大提高了课程教学、解题作业、分析研究的效率。学习掌握 MATLAB，也可以说是在科学计算工具上与国际接轨。

MATLAB 语言比较好学，因为它只有一种数据类型，一种标准的输入输出语句，不用“指针”，不需编译，比其他语言少了很多内容。听三四个小时课，上机练几个小时，就可入门了。以后自学也十分方便，通过它的演示(Demo)和求助(Help)命令，人们可以方便地在线学习各种函数的用法及其内涵。

MATLAB 语言的难点是函数较多，仅基本部分就有 700 多个，其中常用的有二三百个，要尽量多记少查，这可以提高编程效率，而且将会终身受益。

1.2 MATLAB 语言的特点

MATLAB 语言有以下五个特点。

1. 起点高

(1) 每个变量代表一个矩阵，它有 n×m 个元素。从 MATLAB 名字的来源可知，它以矩阵运算而见长，在当前的科学计算中，几乎无处不用矩阵运算，这使它的优势得到了充分的体现。

(2) 每个元素都看作复数。这个特点在其他语言中也是不多见的。

(3) 所有的运算都对矩阵和复数有效，包括加、减、乘、除、函数运算等。

2. 人机界面适合科技人员

(1) 语言规则与笔算式相似。MATLAB 的程序与科技人员的书写习惯相近，因此易写易读，易于在科技人员之间交流。

(2) 矩阵行列数无需定义。要输入一个矩阵，用其他语言时必须先定义矩阵的阶数，而 MATLAB 则不必用阶数定义语句。输入数据的行列数就决定了它的阶数。

(3) 键入算式立即得结果，无需编译。MATLAB 是以解释方式工作的，即它对每条语句解释后立即执行，若有错误也立即作出反应，便于编程者马上改正。这些都大大减少了编程和调试的工作量。

3. 强大而简易的作图功能

(1) 能根据输入数据自动确定坐标绘图。

(2) 能规定多种坐标系(极坐标，对数坐标等)。

(3) 能绘制三维坐标中的曲线和曲面。

(4) 可设置不同颜色、线型、视角等。

如果数据齐全，通常只需一条命令即可出图。

4. 智能化程度高

(1) 绘图时自动选择最佳坐标以及自动定义矩阵维数。

(2) 作数值积分时自动按精度选择步长。

(3) 自动检测和显示程序错误的能力强，易于调试。

5. 功能丰富，可扩展性强

MATLAB 软件包括基本部分和专业扩展两大部分。基本部分包括：矩阵的运算和各种变换，代数和超越方程的求解，数据处理和傅里叶变换，数值积分等，可以充分满足大学理工科本科的计算需要。本书将介绍这部分的主要内容。

扩展部分称为工具箱。它实际上是用 MATLAB 的基本语句编成的各种子程序集，用于解决某一方面的专门问题，或实现某一类的新算法。现在已经有控制系统、信号处理、图像处理、系统辨识、模糊集合、神经元网络、小波分析等 20 余个工具箱，并且它们还在继续发展中。

MATLAB 的核心内容在它的基本部分，所有的工具箱子程序都是用它的基本语句编写的，学好这部分是掌握 MATLAB 必不可少的基础。

1.3　MATLAB 的工作环境

不同版本的 MATLAB 要安装在不同的操作系统下。MATLAB 3.x 之前的版本用的是 DOS 操作系统，MATLAB 4.0 以后的版本都是以 Windows 操作系统为基础的。MATLAB 的工作环境主要由命令窗(Command Window)、图形窗(Figure Window)和文本编辑窗 (File Editor)组成。MATLAB 6.x 和 7.0 又加了几个辅助视窗，组成其“桌面系统”。考虑到 6.x 版本已使用三年半，目前最为普及，而 7.0 版本刚刚推出，本书将以6.x 为典型进行介绍。本章着重介绍命令窗，其他视窗将在第 3 章讨论。

1.3.1　命令窗

在 Windows 桌面上，双击 MATLAB 的图标，就可进入 MATLAB 的工作环境。首先出现 MATLAB 的标志图形，接着出现其缺省的桌面系统，如图 1－1 所示。

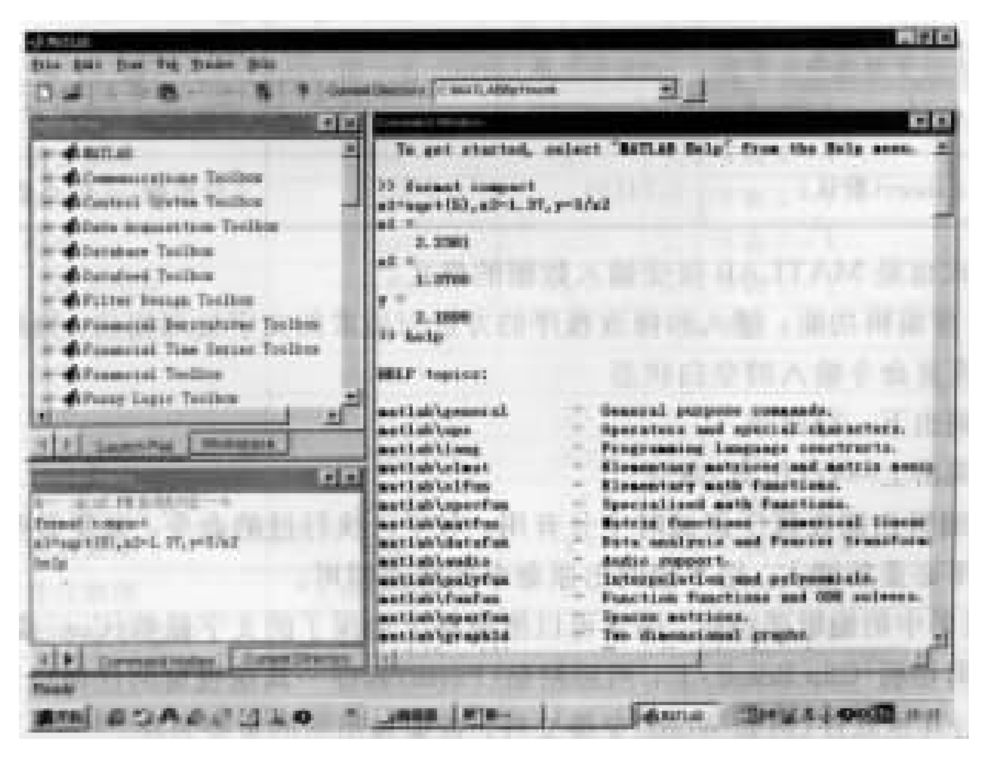


图 1－1　MATLAB 6.x 的桌面系统

其左上视窗为资源目录(Launch Pad)，可切换为工作空间(Workspace)；其左下视窗为历史命令(Command History)，可切换为当前目录(Current Directory)；右半个视窗则为命令窗(Command Window)。命令窗是用户与 MATLAB 进行人机对话的主要环境。＞＞是它的提示符，可以在提示符后键入 MATLAB 的各种命令并读出相应的结果。例如键入

x1＝sqrt(5)，x2＝1.37，y＝3/x2

答案为

x1 = 2.2361　　x2 = 1.3700　　y = 2.1898

命令窗主菜单的有些项目与 Word 相仿，这里只把其中几个主要的做一些说明。

· format 命令：在 MATLAB 默认的 format loose(稀疏格式)下，屏幕上的显示会有许多空行，如果键入 format compact(紧凑格式)，空行就会去掉。format 命令还可以控制数字显示的方式。虽然 MATLAB 是惟一采用双精度格式进行数据的存储和运算的，但数字的显示格式可以有八种。在各格式控制命令下圆周率 π 的显示结果如表 1 - 1 所示。

表 **1** - **1**　数字显示的八种格式

MATLAB 命令	显 示 形 式	说　明
format long	3.14159265358979	16 位十进制数
format short e	3.1416e+000	5 位十进制数加指数
format long e	3.141592653589793e+000	16 位十进制数加指数
format hox	400921fb54442d18	16 位十六进制数
format bank	3.14	2 位小数
format +	+	正、负或零
format rat	355/113	分数近似
format short(默认)	3.14159	2 位整数，4 位小数

显示格式也是 MATLAB 接受输入数据的格式。

· 命令窗编辑功能：键入和修改程序的方法与通常的文字处理相仿。特殊的功能键为

ESC　恢复命令输入的空白状态

↓　调出下一行命令

↑　调出上一行(历史)命令

命令窗编辑功能在程序调试时十分有用。对于已执行过的命令，如要做些修改后重新执行，就可不必重新键入，用 ↑ 键调出原命令做修改即可。

· 主菜单中的编辑项功能：用它可以把屏幕上加深了的文字裁剪(Cut)或复制(Copy)下来，放在剪切板(Clip Board)上，然后粘贴(Paste)到任一其他视窗的任何位置上去。这是 MATLAB 与其他软件(例如 Word)交换文件、数据和图形的重要方法。

· 主菜单中的视图项功能：用它可以改变屏幕上显示的视窗布局。例如，我们希望只显示命令窗，使它占整个屏幕，如图 1 - 2 所示，依次引出 View 的下拉菜单，即【View】→【Desktop Layout】→【Command Window Only】。

· 键入“help”，屏幕上将显示系统中已装入的函数库(即子目录)的名称。如果只装了 MATLAB 的基本部分，则屏幕上将显示出表 1 - 2 中所示的子目录名称。



图 1－2　只显示命令窗的屏幕及其生成的菜单

表 1－2　MATLAB 基本部分的函数库

库　内　容	库　名	库序号	在本书中的章节和表号
数据分析函数库	datafun	(a)	4.1 节表 4－3
动态数据交换库	dde	(g)	3.3 节表 3－4
初等数学函数库	elfun	(c)	2.3 节表 2－7
基本矩阵库	elmat	(d)	2.1 节表 2－1
时间和日期函数库	timefun	(w)	3.1 节表 3－2
非线性数值方法库	funfun	(e)	4.4 节表 4－6
通用命令库	general	(f)	3.1 节表 3－1
数据类型库	datatypes	(b)	4.7 节表 4－11
通用图形函数库	graphics	(h)	2.5 节表 2－13
低层输入/输出函数库	iofun	(j)	3.3 节表 3－3
语言结构函数库	lang	(k)	2.6 节表 2－16
矩阵线性代数库	matfun	(m)	4.2 节表 4－4
运算符和特殊字符库	ops	(n)	2.4 节表 2－10
二维图形库	Graph2d	(p)	2.5 节表 2－12
特殊图形函数库	specgraph	(u)	2.5 节表 2－14
三维图形库	Graph3d	(q)	2.5 节表 2－15
多项式和插值函数库	polyfun	(r)	4.3 节表 4－5
稀疏矩阵函数库	sparfun	(s)	4.6 节表 4－9
特殊数学函数库	specfun	(t)	4.4 节表 4－7
字符串函数库	strfun	(v)	4.5 节表 4－8
用户界面工具库	uitools	(x)	4.7 节表 4－10
MATLAB 演示库	demos	(y)	未列出

※ 键入 help 子目录名，如 help elfun，即得出 elfun 库中各函数名。

※ 键入 help 函数名，如 help tan2，即得到 tan2 函数的意义及用法。

· 退出 MATLAB 有两种方法。一种是键入 exit 或 quit，另一种是用鼠标双击左上角的小方块或单击右上角的×号，后者是非正常退出，该次的输入命令将不记录在“历史命令”中，所以应当尽量避免使用。

1.3.2 图形窗

通常，只要执行了任一种绘图命令，就会自动产生图形窗，以后的绘图都在这一个图形窗中进行。如想再建一个或几个图形窗，则可键入 figure，MATLAB 会新建一个图形窗，并自动给它依次排序。如果要人为规定新图为图 3，则可键入 figure(3)。如要调看已经存在的图形窗 n ，也应键入 figure(n)。

在命令窗中，键入 figure，得出空白的图形窗。如键入 logo，即可生成 MATLAB 的标志图形，如图 1－3 所示。图形窗上的一排按钮，可以用来对图形进行修改或注释。

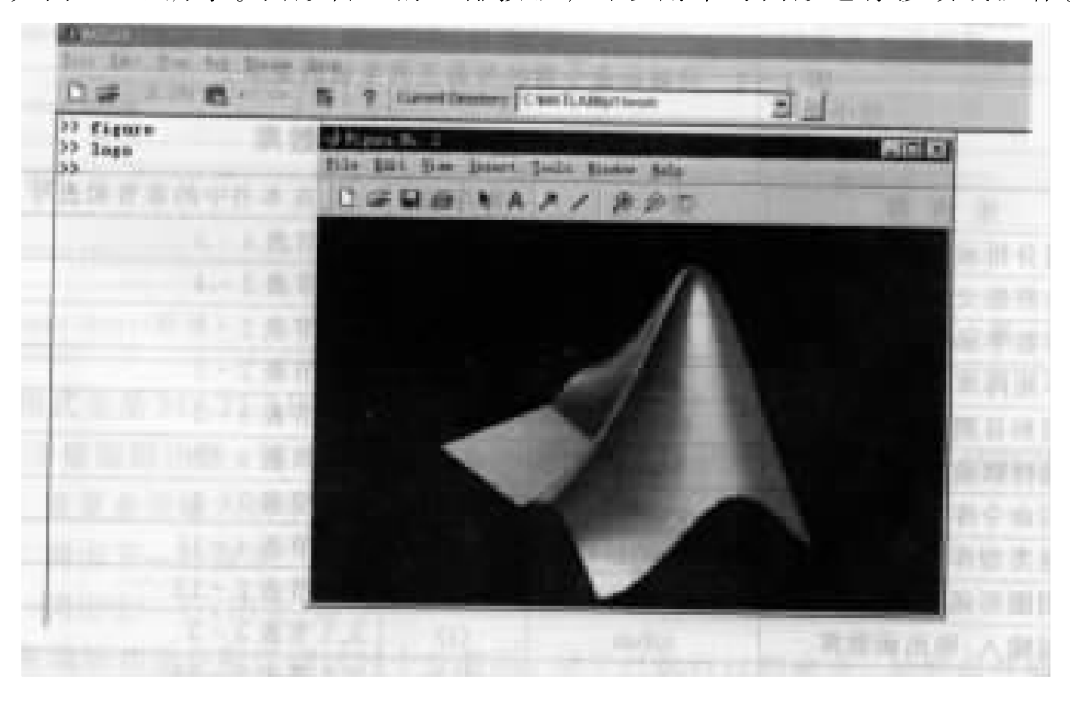


图 1－3 MATLAB 6.x 的命令窗、图形窗和标志图形

1.3.3 文本编辑窗

MATLAB 程序编制有两种方式。一种称为行命令方式，这就是在命令窗中一行一行地输入程序，计算机每次对一行命令作出反应，像计算器那样。这只能编简单的程序，在入门时可以用这种方式。程序稍复杂一些，就应把程序写成一个有多行语句组成的文件，让 MATLAB 来执行这个文件。编写和修改这种文件程序就要用到文本编辑器。

命令窗上方最左边的按钮是用来打开文本编辑器空白页的，左边第二个按钮是用来打开原有程序文件的。打开后的文本编辑窗见图 1－4。

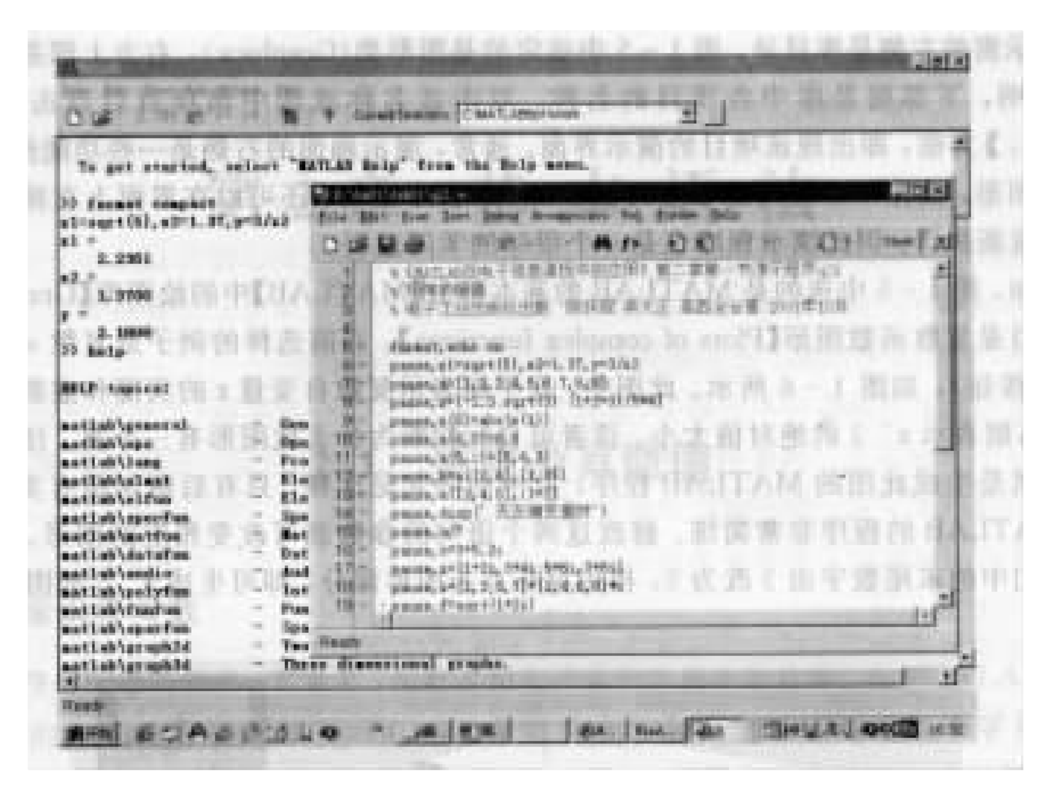


图 1－4　MATLAB 6. x 的命令窗和文本编辑窗

1.4　演示程序

在命令窗中键入 demo，将出现 MATLAB 的演示窗，如图 1－5 所示。

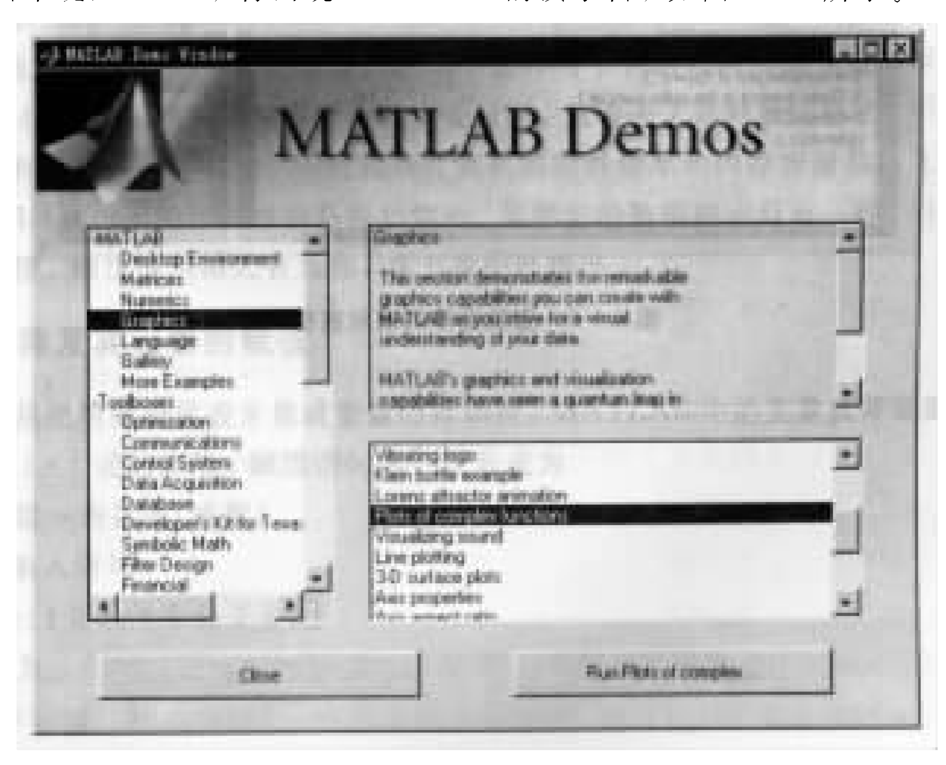


图 1－5　MATLAB 的演示视窗

演示窗的左侧是库目录。图 1 – 5 中选定的是图形类(Graphicx)，右方上部是对该演示库的说明，下部则是库中各项目的名称。双击该名称或选中该项目后点击右下角的【Run ...】方框，即出现该项目的演示界面。通常，演示画面的右侧是一些功能按钮，左上半部是图形，而左下半部则是相应的 MATLAB 程序语句。还可以在界面上直接修改这些语句并重新执行。因此演示程序也是一个很好的学习过程。

例如，图 1 – 5 中选的是 MATLAB 的基本部分【MATLAB】中的绘图库【Graphics】，下选的项目是复数函数图形【Plots of complex functions】。当前选择的例子是复数 z 的三次方(【z^3】按钮)，如图 1 – 6 所示。此图中，底平面表示复数自变量 z 的实部和虚部所张的平面，而高则表示 z^3 的绝对值大小。读者可自行判断为什么此图形有三个翼。注意左下角方框中就是生成此图的 MATLAB 程序，其中前两句是注释，只有后两句是有实效的。这说明 MATLAB 的程序非常简练。修改这两个语句的参数就可改变相应的图形。例如，将最后语句中的末尾数字由 3 改为 5，而用鼠标点击图形部分，即可生成 z^5 的图形。

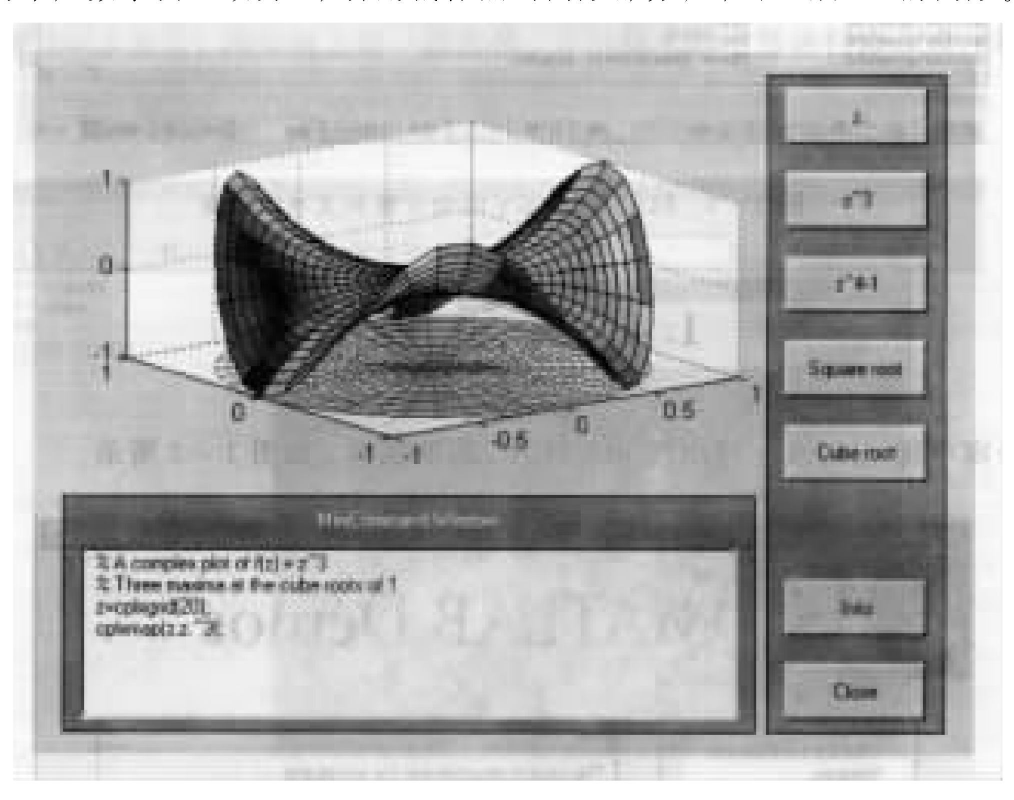


图 1 – 6 复数图形的演示视窗

第 2 章　基 本 语 法

2.1　变量及其赋值

2.1.1　标识符与数

标识符是标志变量名、常量名、函数名和文件名的字符串的总称。在 MATLAB 中，变量和常量的标识符最长允许 19 个字符；函数和文件名则通常不超过八个字符，如果 MATLAB 安装在能管理长文件名的操作系统中，那么，函数名和文件名也可以取到 19 个字符。用于标识符的合法字符包括全部的英文字母(大小写共 52 个)、阿拉伯数字和下划线等符号。标识符中第一个字符必须是英文字母。MATLAB 对大小写敏感(Case Sensitive)，即它把 A 和 a 看作两个不同的字符。

MATLAB 内部只有一种数据格式，那就是双精度(即 64 位)二进制，对应于十进制 16 位有效数和 ±308 次幂。MATLAB 运算和存储时都用双精度格式，这对绝大多数工程计算足够了，许多情况下过于“浪费”。在一些其他的算法语言中设有多种数据格式，如字符型(8 位)、整数型(16 位)、单精度型(32 位)等，这样可节省内存和提高速度，但增加了编程的复杂性。MATLAB 把简化编程作为其主要目标，虽省去了多种数据格式，但也在运算速度和内存消耗方面付出了代价。不过现在计算机的时钟频率和内存容量都以几何级数迅速增长，MATLAB 付出的代价很容易得到弥补。虽然它的数据格式只有一种，但为了人机交互的友好方便，输出显示格式有八种，在上章中已指出。

2.1.2　矩阵及其元素的赋值

赋值就是把数赋予代表常量或变量的标识符。MATLAB 中的变量或常量都代表矩阵，标量应看作 1×1 元的矩阵。赋值语句的一般形式为

变量＝表达式(或数)

例如：输入语句

```
a=[1 2 3;4 5 6;7 8 9]
```

则显示结果为

```
a=1  2  3
  4  5  6
  7  8  9
```

输入

```
x=[-1.3 sqrt(3) (1+2+3)/5*4]
```

结果为

```
x=-1.3000    1.7321    4.8000
```

可以看出，矩阵的值放在方括号中，同一行中各元素之间以逗号或空格分开，不同的行则以分号隔开，语句的结尾可用“回车”或逗号，此时会立即显示运算结果。如果不希望显示结果，就以分号结尾。此时运算仍然执行，只是不作显示。

变量的元素用圆括号中的数字(也称为下标)来注明，一维矩阵(也称数组或向量)中的元素用一个下标表示，两维的矩阵可有两个下标数，以逗号分开。三维和更高维的矩阵，可有三个或更多下标。用户可以单独给元素赋值，如 x(2)=1.7321，a(2，3)=6 等。如果赋值元素的下标超出了原来矩阵的大小，矩阵的行列会自动扩展。例如输入

```
x(5)=abs(x(1))
```

结果为

```
x=-1.3000    1.7321    4.8000    0    1.3000
```

又如键入

```
a(4,3)=6.5
```

得

```
a=1.0000    2.0000    3.0000
  4.0000    5.0000    6.0000
  7.0000    8.0000    9.0000
       0         0    6.5000
```

可见，跳空的元素 x(4)，a(4，1)，a(4，2)被自动地赋值 0。这种自动扩展维数的功能只适用于赋值语句。在其他语句中若出现超维调用的情况，MATLAB 将给出出错提示。

给全行赋值，可用冒号“：”。例如，给 a 的第 5 行赋值，可键入

```
a(5,:)=[5,4,3]
```

得

```
a=1.0000    2.0000    3.0000
  4.0000    5.0000    6.0000
  7.0000    8.0000    9.0000
       0         0    6.5000
  5.0000    4.0000    3.0000
```

把 a 的第 2、4 行及 1、3 列交点上的元素取出，构成一个新矩阵 b，可键入

```
b=a([2,4],[1,3])
```

得

```
b=4.0000    6.0000
       0    6.5000
```

要抽去 a 中的 2、4、5 行，可利用空矩阵[ ]的概念。键入

```
a([2,4,5],:)=[ ]
```

得到

```
a=1    2    3
  7    8    9
```

注意，“空矩阵”是指没有元素的矩阵。对任何一个矩阵赋值为[ ]，就是使它的元素都消失掉。这完全不同于“零矩阵”，后者是元素存在，只是其数值为零而已。可以看出，空矩阵在使矩阵减缩时是不可缺少的概念。

除“变量＝表达式(或数)”的标准形式外，可以不要等式左端而只剩下“表达式”。这有两种可能：① 该表达式并不产生数字解，例如产生图形、或改变系统状态；② 该表达式产生数字解，但不需保存它。此时 MATLAB 自动给出一个临时变量 ans，把右端的结果暂存在 ans 中。若再作下一次运算又用到 ans，则前一次的结果就被冲销了。例如键入

```
a/7
```

得到

```
ans=    0.1429    0.2857    0.4286
        1.0000    1.1429    1.2857
```

2.1.3 复数

MATLAB 中的每一个元素都可以是复数，实数是复数的特例。复数的虚数部分用 i 或 j 表示。这是在 MATLAB 启动时就已在内部设定的。例如，键入

```
c=3+5.2i
```

得

```
c=3.0000+5.2000i
```

对复数矩阵有两种赋值方法，既可将其元素逐个赋予复数，例如，键入

```
z=[1+2i, 3+4i; 5+6i, 7+8i]
```

得

```
z= 1.0000+2.0000i    3.0000+4.0000i
   5.0000+6.0000i    7.0000+8.0000i
```

也可将其实部和虚部矩阵分别赋值，如

```
z=[1, 3; 5, 7]+[2, 4; 6, 8]*i
```

两种赋值方法得出同样结果。注意只有数字和 i 的乘积式中可省略乘号，在上述矩阵式中若省略乘号“*”，就会出错。另外，如果在前面程序中曾经给 i 或 j 赋过其他值，则 i，j 已经不是虚数符号，这些虚数赋值语句都不对了。此时应键入

```
clear i, j
```

即把原有的 i，j 清掉，然后再执行复数赋值语句。

MATLAB 中所有的运算符和函数都对复数有效。例如键入

```
f=sqrt(1+2i)
```

得

```
f=1.2720+0.7862i
```

检验

```
f*f
ans=1.0000+2.0000i
```

因此，复数的表达式同样都能作为赋值语句。再来看复数矩阵 z 的转置、共轭运算：运算符“′”表示把矩阵作共轭转置，即把它的行列互换，同时把各元素的虚部反号。函数 conj 则只把各元素的虚部反号，即只取共轭。所以若想求转置而不要共轭，就要把 conj 和“′”结合起来完成。键入

```
w=z', u=conj(z), v=conj(z)'          %共轭转置，共轭和转置
```

得

```
w=1.0000-2.0000i     5.0000-6.0000i
  3.0000-4.0000i     7.0000-8.0000i
u= 1.0000-2.0000i    3.0000-4.0000i
   5.0000-6.0000i    7.0000-8.0000i
v= 1.0000+2.0000i    5.0000+6.0000i
   3.0000+4.0000i    7.0000+8.0000i
```

2.1.4 变量检查

在调试程序时，往往需要检查工作空间中的变量及其维数。可用 who 或 whos 命令：键入

```
who
```

得

```
Your variables are:
a       c       v       x1       z
ans     f       w       x2
b       u       x       y
```

这些就是我们前面用过的变量，如果还需要知道它们的详细特征，可键入 whos，

结果为：

变量名	维数	元素数	字节数	密度	复数
a	2 by 3	6	48	Full	No
ans	2 by 2	4	64	Full	Yes
b	2 by 2	4	32	Full	No
c	1 by 1	1	16	Full	Yes
…	…	…	…	…	…
z	2 by 2	4	64	Full	Yes

Grand total is 40 elements using 496 bytes(共 40 个元素，占 496 字节)

可以看出，每个实元素占八个字节，复元素则占 16 个字节。读者可自行解释其原因。

MATLAB 中实际上还有几个内定的变量，在变量检查时不作显示。把它们列于表 2-1 中。这里着重介绍一下 Inf 和 NaN。

Inf(还有-Inf)是无穷大，键入 1/0 就可得到它。NaN 是非数(Not a Number)的缩写，由 0/0，0 * Inf 或 Inf/Inf 而得。在其他语言中遇到上述非法运算时，系统就停止运算并退出。而 MATLAB 却不停止运算，仍给结果赋于 Inf 或 NaN，并继续把程序执行完。这有很大的好处，可以避免因为一个数据不好而破坏全局。出现 Inf 或 NaN 后，对它们作的任何运算，结果仍为 Inf 或 NaN，这种运算规则称为 IEEE(电工和电子工程师协会)运算规则，是 IEEE 的一种标准。

表 2－1 基本矩阵和矩阵运算(elmat)(d)

基本矩阵	zeros	全零矩阵(m×n 维)	logspace	对数均分向量(1×n 维数组)
	ones	全幺矩阵(m×n 维)	Freqspace	频率特性的频率区间
	rand	随机数矩阵(m×n 维)	meshgrid	画三维曲面时的 X，Y 网格
	randn	正态随机数矩阵(m×n 维)	meshdom	在 MATLAB5 中取消
	eye(n)	单位矩阵(方阵)	:	将元素按列取出排成一列
	linspace	均分向量(1×n 维数组)		
特殊变量和函数	ans	最近的答案	inf	Infinity(无穷大)
	eps	浮点数相对精度	NaN	Not-a-number(非数)
	realmax	最大浮点实数	flops	浮点运算次数
	realmin	最小浮点实数	computer	计算机类型
	pi	3.141 592 353 585 79	inputname	输入变量名
	i，j	虚数单位	size	多维矩阵的各维长度
	length	一维矩阵的长度		
矩阵结构提取和变换	cat	链接数组	diag	提取或建立对角阵
	fliplr	矩阵左右翻转	ind2sub	把元素序号变为矩阵下标
	flipud	矩阵上下翻转	sub2ind	把矩阵下标变为元素序号
	repmat	复制和排成矩阵	tril	取矩阵的左下三角部分
	reshape	维数重组(元素总数不变)	triu	取矩阵的右上三角部分
	Rot90	矩阵整体反时针旋转 90°		
特殊矩阵	compan	Companion 矩阵	magic	魔方矩阵
	gallery	Higham 测试矩阵	pascal	Pascal 矩阵
	hadamard	Hadamard 矩阵	rosser	经典的对称特征值测试问题
	hankel	Hankel 矩阵	toeplitz	Toeplitz 矩阵
	hilb	Hilbert 矩阵	vander	Vandermonde 矩阵
	invhilb	Hilbert 逆矩阵	wilkinson	Wilkinson's 特征值测试矩阵

2.1.5 基本赋值矩阵

为了给大量元素赋值的方便，MATLAB 提供了一些基本矩阵。表 2－1 给出了最常用的一些。其用法可从下面的例子中看到。其中魔方矩阵 magic(n) 的特点是：其元素由 1 到 n×n 的自然数组成；每行、每列及两对角线上的元素之和均等于 $(n^3+n)/2$。单位矩阵 eye(n)是 n×n 元的方阵，其对角线上的元素为 1，其余元素均等于 0。下例列出了上表中的四种矩阵：

键入

f1＝ones(3，2)，f2＝zeros(2，3)，f3＝magic(3)，f4＝eye(2)

得幺矩阵

```
f1=1    1
   1    1
   1    1
```

零矩阵

```
f2=0    0    0
   0    0    0
```

魔方矩阵

```
f3=8    1    6
   3    5    7
   4    9    2
```

单位矩阵

```
f4=1    0
   0    1
```

线性分割函数 linspace(a，b，n)在 a 与 b 之间均分地产生 n 个点值，形成 1×n 元向量。例如，键入

f5＝linspace(0，1，5)

得

```
f5=   0    0.2500    0.5000    0.7500    1.0000
```

大矩阵可由若干个小矩阵组成，但必须其行列数正确，恰好填满全部元素。如键入

fb1＝[f1，f3；f4，f2]

得

```
fb1=1    1    8    1    6
    1    1    3    5    7
    1    1    4    9    2
    1    0    0    0    0
    0    1    0    0    0
```

可再由它与 f5 组成一个更大的矩阵，键入

fb2＝[fb1；f5]

得

```
fb2=1.0000    1.0000    8.0000    1.0000    6.0000
    1.0000    1.0000    3.0000    5.0000    7.0000
    1.0000    1.0000    4.0000    9.0000    2.0000
    1.0000    0         0         0         0
    0         1.0000    0         0         0
    0         0.2500    0.5000    0.7500    1.0000
```

可以看出，fb1 和 fb2 显示的数据不同，fb1 的元素都是整数，而 fb2 则都是带小数的。其原因是 MATLAB 要求一个矩阵中所有元素用同一显示格式。因为 f5 中元素含小数，所以所有的元素都得用小数格式来显示(0 元素除外，它表示真正的 0 与无法显示的小数值 0.0000 不同)。

为了用同一显示格式，当矩阵中的最大元素小于 0.001 或其最小元素大于 1000 时，

MATLAB会把其中小于0.001或大于1000的公因子提出来，如键入由两个很小的数组成的矩阵

f=[0.000073，　5.33e-6]

得到　f=1.0e-004 *

0.7300　　0.0533

如果矩阵中出现大小差别很多的元素，则显示时将以大元素优先，小元素就只能显示很少的有效位，甚至成为0.0000。这时不要误以为它是0，可以用显示单个元素的命令来得到它的准确值，也可改用长格式(Format Long)来显示整个矩阵。

2.2　矩阵的初等运算

2.2.1　矩阵的加减乘法

矩阵算术运算的书写格式与普通算术运算相同，包括加、减、乘、除。也可用括号来规定运算的优先次序，但它的乘法定义与普通数(标量)不同。相应地，作为乘法逆运算的除法也不同，有左除(\)和右除(/)两种符号。

两矩阵的相加(减)就是其对应元素的相加(减)，因此，该两矩阵的阶数必须相同，且它们的和也有相同的阶数。检查矩阵阶数的MATLAB语句是size，例如，键入

[n，m]=size(fb2)

得　　n=6　　m=5　　(6行5列)

如果要自己编写矩阵A和B相加(减)的程序，就必须先求nA，mA，nB，mB，并检验是否满足nA=nB和mA=mB。确认无误后再按对应元素相加(减)，得出C=A+B(或CA-B)。如果阶数检验不合格，则显示出错。当两个相加矩阵中有一个是标量时，MATLAB承认算式有效，它自动把该标量扩展成同阶等元素矩阵，与另一矩阵相加。例如，键入

X=[-1 0 1]；Y=X-1

得　　Y=　-2　　-1　　0

如果已经知道X是一维矩阵(数组)，也可以用

l=length(X)

来求它的长度。注意size有两个输出量，而length只有一个输出量。length不区分列或行，所以作加减法阶数检验时只能用size。

现在来看矩阵的乘法。n×p阶矩阵A与p×m阶矩阵B的乘积C是一个n×m阶矩阵，它的任何一个元素C(i，j)的值为A阵的第i行和B阵的第j列对应元素乘积的和。即

$$C(i,j) = A(i,1) * B(1,j) + A(i,2) * B(2,j) + \cdots + A(i,p) * B(p,j)$$
$$= \sum_{p} A(i,k) * B(k,j)$$

式中的乘号是普通数(标量)的乘号。p是A阵的列数，也是B阵的行数，也称为两个相乘矩阵的内阶数，两矩阵相乘的必要条件是它们的内阶数相等。不难看出，对于标量A，B，因为n，p，m均为1，矩阵乘法就退化为普通数的乘法。

如果要自己编写矩阵 A 和 B 相乘的程序，就必须先求 nA，mA，nB，mB，并检验 mA 是否等于 nB，确认无误后再按上式把对应元素相乘后累加，得出 C(i，j)。分别取 i 从 1 到 nA，j 从 1 到 mB，得出 nA×mB 个 C 元素，排成矩阵形式，得到 C。

实际 MATLAB 已将上述矩阵加、减、乘的程序编程为内部函数，我们只要用+、-、* 作运算符号就包含了检查阶数和执行运算的全过程，而且其运算对复数有效。

由此可见，前面定义的 X 和 Y 是不能相乘的，因为它们的内阶数分别为 3 和 1。与加减法相仿，如果乘数中的一个是标量，则 MATLAB 不检查其内阶数，而用该标量乘以矩阵的每个元素。例如，键入

```
pi * X
```

得
```
ans= -3.1416      0      3.1416
```

若把 Y 转置，成为 3×1 阶，内阶数变为与 X 相同，就可求

```
X * Y'
```

得
```
ans=2
```

不难用心算检验其正确性。这个式子可读成 X 左乘 Y′。现在让 X 右乘 Y′，这时两者的内阶数都是 1，而外阶数(定义中的 nA 和 mB)都成了 3。于是有

```
Y' * X
```

得
```
ans=  2     0     -2
      1     0     -1
      0     0      0
```

显然，X 左乘和右乘 Y′所得的结果是完全不同的。只有单位矩阵例外，单位矩阵乘以任何矩阵 A(其阶数为 nA * mA)时，不管是左乘还是右乘，其积仍等于该矩阵。即

```
eye(nA) * A=A
A * eye(mA)=A
```

读者可按单位矩阵的定义自行检验其正确性。

设有三个线性方程组成的联立方程组：

$$x_1+2x_2+3x_3=2$$
$$3x_1-5x_2+4x_3=0$$
$$7x_1+8x_2+9x_3=2$$

令　a=[1，2，3；3，-5，4；7，8，9]；$x=[x_1, x_2, x_3]$；b=[2；0；2]

则这个联立方程组就可表为简洁的矩阵形式

```
a * X'= b
```

给出 a 和 X，可以用矩阵乘法求出 b。但若给出 a 和 b，要求 X，那就需要逆运算——除法了。

2.2.2 矩阵除法及线性方程组的解

在线性代数中，本来就没有除法，只有“逆矩阵”。矩阵除法是 MATLAB 从逆矩阵的概念引申来的。先介绍逆矩阵的定义，对于任意 n×n 阶方阵 A，如果能找到一个同阶的方阵 V，使

```
A * V=I
```

其中，I 为 n 阶的单位矩阵 eye(n)，则 V 就是 A 的逆阵。数学符号表示为

$$V=A^{-1}$$

逆阵 V 存在的条件是 A 的行列式 det(A)不等于 0，V 的最古典求法为高斯消去法，在线性代数中介绍。在 MATLAB 中已经做成了内部函数 inv，键入

V＝inv(A)

就可得到 A 的逆矩阵 V。如果 det(A)等于或很接近于零，则 MATLAB 会显示出错或警告信息：“A 矩阵病态(ill-conditioned)，结果精度不可靠。”

现在来看方程 D＊X＝B，设 X 为未知矩阵，在等式两端同左乘以 inv(D)，即

inv(D)＊D＊X ＝ inv(D)＊B

因为等式左端 inv(D)＊D＝I，而 I＊X＝X，上式成为

X＝inv(D)＊B ＝ D B

把 D 的逆阵左乘以 B，MATLAB 就记作 D、称之为“左除”。从 D＊X＝B 的阶数检验可知，B 与 D 的行数相等，因此，左除时的阶数检验条件是：两矩阵的行数必须相等。

如果原始方程的未知矩阵在左而系数矩阵在右，即

X＊D＝B

则按上述同样的方法可以写出

X＝B＊inv(D)＝B/D

把 D 的逆阵右乘以 B，我们就记作 D/，称之为“右除”。同理，右除时的阶数检验条件是：两矩阵的列数必须相等。

下面来看矩阵左右乘除的一些示例。设 A＝[1，2，3；4，5，6]，B＝[2，4，0；1，3，5]，D＝[1，4，7；8，5，2，；3，6，0]

其乘除的结果列于表 2－2 中。

表 **2－2** 矩阵乘除法的示例

算 式	答 案
A＊B	??? Error using ＝＝＞ ＊ Inner matrix dimensions must agree.(内阶数必须相等)
A′＊B	6 16 20 9 23 25 12 30 30
A＊B′	10 22 28 49
D A	??? Error using ＝＝＞ Matrix dimensions must agree.(矩阵阶数应相符)
D A′	－0.0370 0 0.5185 1.0000 －0.1481 0.0000
A/D	0.4074 0.0741 0.0000 0.7407 0.4074 0.0000

矩阵除法可以用来方便地解线性方程组。例如要求下列方程组的解 $x=[x_1; x_2; x_3]$：

$$6x_1+3x_2+4x_3=3$$

$$-2x_1+5x_2+7x_3=-4$$

$$8x_1-4x_2-3x_3=-7$$

此式可写成矩阵形式 A * x＝B，求解的 MATLAB 程序为

```
Λ=[6,3,4;-2,5,7;8,-4,-3];   B=[3;-4;-7];   x=Λ  B
```

得

```
x=   0.6000
     7.0000
    -5.4000
```

MATLAB 中的除法还可以用来解方程数不等于未知数个数的情况。比如再加上一个方程

$$x_1+5x_2-7x_3=9$$

这时系数矩阵 A1 成为 4×3 阶，不难看出 A1 的行数 nA1 是方程数，其列数 mA1 是未知数的个数，nA1＞mA1，说明方程组是超定的，方程无解。我们照样列 MATLAB 程序

```
A1=[6,3,4;-2,5,7;8,-4,-3;1,5,-7];B1=[3;-4;-7;9];
x1=A1  B1
```

答案为

```
x1=-0.1564
     1.0095
    -0.6952
```

它并未显示出错信息，却给出了解，这怎么可能呢？实际上这时 MATLAB 给出的是最小二乘解。把这个 x1 代入方程组，肯定任何一个方程都不满足并都可得出一个误差，把这四个误差的平方相加开方，称为均方差。解 x1 保证比其他任何解所得的均方差都小。

对方程数少于未知数个数的情况：A1 矩阵的 nA1＜mA1，说明方程组是不定的，它有无穷个解。此时仍然可用除法符号来求出解。这个解是满足方程的，但它不是惟一解。它是令 x1 中某个或某些元素为零的一个特殊解。

2.2.3　矩阵的乘方和超越函数

MATLAB 的乘幂函数"^"、指数函数 expm、对数函数 logm、和开方函数 sqrtm 是对矩阵进行的，即把矩阵作为一个整体来运算。除此之外，其他 MATLAB 函数都是对矩阵中的元素分别进行，英文直译为数组运算(Array Operations)，较准确的意义应为"元素群运算"，这将在下一节讨论。

在幂次运算时矩阵可以作为底数，指数是标量。这是矩阵乘法的扩展，为了保证作乘法时内阶数相同，矩阵作为底数时，它必须是方阵。矩阵也可以作为指数，底数是标量，它仍然应是方阵。底数和指数不能同时为矩阵，如果这样输入，将显示出错信息。表 2-3 给出了一些语句及其结果。注意 sqrtm 与 sqrt、expm 与 exp 以及 logm 与 log 函数的不同，即矩阵整体运算和矩阵元素群运算的不同。

表 2-3 矩阵整体运算的例及其结果

$$\left(\text{此表中 } s=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix},\ D=\begin{bmatrix}1 & 4 & 7\\ 8 & 5 & 2\\ 3 & 6 & 0\end{bmatrix}\right)$$

键入语句	输出结果	说明
D^2	54 66 15 54 69 66 51 42 33	按矩阵运算
2^D	2 16 128 256 32 4 8 64 1	按元素运算
D^s	??? Error using ==> ^ Matrix dimensions must agree.	非法运算
u1=sqrtm(s)	u1= 0.5537+0.4644i 0.8070-0.2124i 1.2104-0.3186i 1.7641+0.1458i	按矩阵运算 可用 u1 * u1=s 检验
u2=sqrt(s)	u2= 1.0000 1.4142 1.7321 2.0000	按元素运算，u2 * u2≠s u2. * u2=s(见下节)
v1=expm(s)	v1= 51.9690 74.7366 112.1048 164.0738	按矩阵运算 可用 logm(v1)=s 检验
v2=exp(s)	v2= 2.7183 7.3891 20.0855 54.5982	按元素运算 可用 log(v1)=s 检验
logm(D)	1.2447 -0.9170 2.8255 1.6044 2.5760 -1.9132 -0.7539 1.1372 1.6724	按矩阵运算
log(D)	0 1.3863 1.9459 2.0794 1.6094 0.6931 1.0986 1.7918 -Inf	按元素运算

2.2.4 矩阵结构形式的提取与变换

在作矩阵运算时，往往需要提取其中的某些特殊结构的元素来组成新的矩阵；有时则要改变矩阵的排列。除了我们在 2.1 节讲过的提取行、列和将行列转置的语句之外，MATLAB 还提供了一些改变矩阵结构的函数。这些函数列于表 2-4 中，同时给出了相应的例子。

表 2－4 矩阵结构形式提取和变换语句

$$\text{设 } A=\begin{bmatrix} 8 & 1 & 6 & 0 \\ 3 & 5 & 7 & 1 \\ 4 & 9 & 2 & 2 \end{bmatrix}$$

函数名	功 能 说 明	语 句	结 果
fliplr	矩阵左右翻转	B＝fliplr(A)	B＝ 0 6 1 8 1 7 5 3 2 2 9 4
flipud	矩阵上下翻转	B＝flipud(A)	B＝ 4 9 2 2 3 5 7 1 8 1 6 0
reshape	阶数重组(元素总数不变)	B＝reshape(A，2，6)	B＝ 8 4 5 6 2 1 3 1 9 7 0 2
Rot90	矩阵整体反时针旋转 90°	B＝Rot90(A)	B＝ 0 1 2 6 7 2 1 5 9 8 3 4
diag	提取或建立对角阵	B＝Diag(A)	B＝ [8，5，2]
tril	取矩阵的左下三角部分	B＝tril(A)	B＝ 8 0 0 0 3 5 0 0 4 9 2 0
triu	取矩阵的右上三角部分	B＝triu(A)	B＝ 8 1 6 0 0 5 7 1 0 0 2 2
：	将元素按列取出排成一列	B＝A(：)′	B＝8 3 4 1 5 9 6 7 2 0 1 2

2.3 元 素 群 运 算

元素群运算能大大简化编程，提高运算的效率，是 MATLAB 优于其他许多语言的一个特色。

2.3.1 数组及其赋值

数组通常是指单行或单列的矩阵，一个 N 阶数组就是 1×N 或 N×1 阶矩阵。一个 N 阶数组可以表述一个 N 维向量。因为一个三维空间的向量可用它在三个坐标上的投影来表示，即 v＝[vx，vy，vz]，也即表示为三阶的数组。这个概念也可扩展到 N 维空间的向量。

在求某些函数值或曲线时，常常要设定自变量的一系列值，例如设时间 t 从 0～1 秒之间，每隔 0.02 秒取一个点，共 51 个点，这是 1×51 阶的数组。如果逐点给它赋值，将非常麻烦。MATLAB 提供了两种为等间隔数组赋值的简易方法。

（1）用两个冒号组成等增量语句，其格式为：t=[初值：增量：终值]。如键入

t=[0：0.02：1]

得　　t= 0　0.020　0.040　…　0.960　0.980　1.000

此语句中的增量也可设为负值，此时初值要比终值大。如键入

z=10：-3：-5

得　　z= 10　7　4　1　-2　-5

当增量为 1 时，这个增量值可以略去，因而该语句只有一个冒号。如键入

k=1：6

得　　k= 1　2　3　4　5　6

（2）用 linspace 函数。如键入

theta=linspace(0，2 * pi，9)

得　　theta=0　0.7854　1.5708　2.3562　3.1416　3.9270　4.7124　5.4978　6.2832

在圆周上 0 和 2 * pi 实际上是一个点，所以这个命令是把圆周分为八份。

有时要求自变量按等比级数赋值，在设频率轴时往往如此，这时可用 logspace 函数。如键入

w=logspace(0，1，11)

得　　w=1.0000　1.2589　1.5849　…　6.3096　7.9433　10.0000

它的意义是从 10 的 0～1 次幂之间按幂等分(即数是等比的)为 11 个点。

2.3.2　元素群的四则运算和幂次运算

元素群运算也就是矩阵中所有元素按单个元素进行运算。为了与矩阵作为整体的运算符号相区别，要在运算符“ * 、/、\、^”前加一点符号“.”，以表示进行的是元素群运算。参与元素群运算的两个矩阵必须是同阶的(标量除外，它会自动扩展为同阶矩阵参与运算)，表 2-5 中表示两个 1×3 阶的元素群运算的结果。因为数取得很简单，可用心算加以检验。元素群的幂次运算也就是各个元素自行作幂次运算，对每个元素而言，这种运算和对标量运算一样，所以很容易判定它的正确性。

表 2-5　简单的元素群运算

(设 X=[1，2，3]；Y=[4，5，6])

运　算　式	输　出　结　果	说　　明
Z=X. * Y	Z=4　10　18	思考问题：X * Y 能成立吗？
Z=X.　Y	Z=4.0000　2.5000　2.0000	元素群没有左除右除之分
Z=X.^Y	Z=1　32　729	思考问题：X^Y 能成立吗？
Z=X.^2	Z=1　4　9	思考问题：X^2 能成立吗？
Z=2.^[X Y]	Z=2　4　8　16　32　64	思考问题：2^[X Y]能成立吗？

表 2-6 中参与元素群运算的是一个 3×3 阶的方阵，这是为了便于比较矩阵中的元素运算和矩阵整体运算的不同(因为非方阵是不能按整体作矩阵乘幂运算的)。从中也可以看出，不宜元素群运算称为数组运算，因为这里参加运算的是一个 3×3 阶矩阵而不是数组。

表 2－6 元素群和矩阵幂次运算的比较

$$\left\{ 设\ D=\begin{bmatrix}1 & 4 & 7\\ 8 & 5 & 2\\ 3 & 6 & 0\end{bmatrix} \right\}$$

输入算式	D			D^3			D.^3			3.^D		
输出结果	1	4	7	627	636	510	1	64	343	3	81	2187
	8	5	2	804	957	516	512	125	8	6561	243	9
	3	6	0	486	612	441	27	216	0	27	729	1

特别注意 D^3 和 D.^3 结果的不同。3.^D 与 3^D 也完全不同，3^D 是合法运算，读者可自行实践并对其结果作出解释，不过这是比较难的题目，本书不作要求。

2.3.3 元素群的函数

前已指出，所有的 MATLAB 函数都适用于作元素群运算，只有专门说明的几个除外。那就是 *、/、\、^ 运算符和 sqrtm、expm、logm 三个函数。表 2－7 基本函数库中的常用函数都可用于元素群运算，也即其自变量都可以是任意阶的矩阵。

下面的例子可以说明利用元素群运算的优越性。例如，要求列出一个三角函数表，这在 MATLAB 中只要两条语句，键入

```
x=[0：0.1：pi/4]'；[x，sin(x)，cos(x)，tan(x)]
```

第一条语句把数组 x 赋值，经转置后成为一个列向量。因为 sin，cos，tan 函数都对元素群有效，得出的都是同阶的列向量。第二条语句把四个列向量组成一个矩阵，并进行显示。得

```
     0         0       1.0000       0
  0.1000    0.0998    0.9950    0.1003
  0.2000    0.1987    0.9801    0.2027
  0.3000    0.2955    0.9553    0.3093
  0.4000    0.3894    0.9211    0.4228
  0.5000    0.4794    0.8776    0.5463
  0.6000    0.5646    0.8253    0.6841
  0.7000    0.6442    0.7648    0.8423
```

第一列是 x，以下各列依次是 sin(x)，cos(x)，tan(x)。如果要加一个表头，第二条语句可改成两条如下的显示语句：

```
disp('      x      sin(x)      cos(x)      tan(x) ')
disp([x，sin(x)，cos(x)，tan(x)])
```

disp 后括号内引号中的内容是直接显示的，放入空格就显示空格，放入汉字就显示汉字。后一句括号中没有引号，是变量名组成的矩阵，它就显示该矩阵中各变量的值。

表 2－7　基本函数库(elfun)(未标注的为单输入单输出函数)(c)

三角函数	sin	正弦	cos	余弦
	tan	正切	asin	反正弦
	acos	反余弦	atan	反正切
	atan2(x, y)	四象限反正切	sinh	双曲正弦
	cosh	双曲余弦	tanh	双曲正切
	acosh	反双曲余弦	atanh	双曲正切
	asinh	反双曲正弦	sec	正割
	csc	余割	cot	余切
	asec	反正割	acsc	余割
	acot	反余切	sech	双曲正割
	csch	双曲余割	coth	双曲正切
	asech	反双曲正割	acsch	反双曲余割
	acoth	反双曲正切		
指数函数	exp	以 e 为底的指数	log	自然对数
	log2	以 2 为底的指数	log10	以 10 为底的对数
	pow2	2 的幂	sqrt	方根
	nextpow2	比输入数大而最近的 2 的幂		
复数	abs	绝对值和复数模值	angle	相角
	real	实部	imag	虚部
	conj	共轭复数	isreal	是实数时为真
	unwrap	去掉相角突变	cplxpair	按复数共轭对排序元素群
取整函数	round	四舍五入为整数	fix	向 0 舍入为整数
	floor	向－∞舍入为整数	ceil	向∞舍入为整数
	sign	符号函数	rem(a, b)	a 整除 b，求余数
	mod(x, m)	x 整除 m 取正余数		

2.4　逻辑判断及流程控制

2.4.1　关系运算

所谓关系运算是指两个元素之间数值的比较，一共有六种可能。

<	<=	>	>=	==	~=
小于	小于等于	大于	大于等于	等于	不等于

关系运算的结果只有两种可能，即 0 或 1。0 表示该关系式为“假”，即它不成立；1 表示该关系式为“真”，即该关系式成立。例如键入关系式

```
a=2+2==4
```

得

```
a=1
```

注意，前面的单个等号表示赋值，后面的双等号则表示关系运算。式中 2+2==4 是关系运算，它的优先级高，算出的结果给 a 赋值，为了改善可读性，最好加上括号，写成 a=(2+2==4)，表明是把括号内的关系式的结果给 a 赋值。但表示关系运算的相等符号仍必须用双等号。

MATLAB 中的关系运算都适用于矩阵，它是对矩阵的各个元素进行元素群运算，因此两个相比较的矩阵必须有相同的阶数，输出的结果也是同阶矩阵。例如键入

```
A=magic(6)
```

得

```
A=  35    1    6   26   19   24
     3   32    7   21   23   25
    31    9    2   22   27   20
     8   28   33   17   10   15
    30    5   34   12   14   16
     4   36   29   13   18   11
```

要找到此矩阵中所有被 3 整除的元素，并在其位置上标以 1。可以用表 2-7 中的 rem 函数，rem(A，3)表示把 A 除以 3 的余数，余数为零就是整除。

键入

```
p=(rem(A，3)==0)
```

得

```
p=  0    0    1    0    0    1
    1    0    0    1    0    0
    0    1    0    0    1    0
    0    0    1    0    0    1
    1    0    0    1    0    0
    0    1    0    0    1    0
```

关系运算中还包括某些条件判断，例如判断矩阵元素中有无 NaN、Inf 值，矩阵是否为实数阵、稀疏阵或空阵等，它们不能直接用上述六种关系符简单地表述，MATLAB 把它们编成了专用的函数以备直接调用，见表 2-10。

[j，k]=find(p) 给出 p 矩阵中不为零的元素的两个下标，find(p)或 lp=find(p)给出 p 矩阵中不为零的元素的序号。矩阵元素是按列排序号的，先第一列，再接第二列……依次排完后，再确定它们的顺序号。一个 6×6 的矩阵的 36 个元素的序号排列见表 2-8。因此一个 n×m 阵中下标为(j，k)的元素，其序号为 l=(k-1)*n+j。

键入

```
lp=find(p)'
```

得

```
lp=  2    5    9   12   13   16   20   23   27   30   31   34
```

可以看出这些序号确实对应于 p 中的 1 元素。矩阵的序号(index)与下标(subscript)是

一一对应的，其相互变换关系可由表 2－1 中的 ind2sub（读作 index to subscirpt）和 sub2ind 函数求得。

表 2－8 矩阵元素的序号排法

1	7	13	19	25	31
2	8	14	20	26	32
3	9	15	21	27	33
4	10	16	22	28	34
5	11	17	23	29	35
6	12	18	24	30	36

2.4.2 逻辑运算

逻辑量只能取 0（假）和 1（真）两个值。逻辑量的基本运算为“与（&）”、“或（|）”和“非（～）”三种。有时也包括“异或（xor）”，不过“异或”可以用三种基本运算组合而成。两个逻辑量经这三种逻辑运算后的输出仍然是逻辑量，表示逻辑量的输入/输出关系的表称为真值表，见表 2－9。

表 2－9 基本逻辑运算的真值表

运　算	A＝0		A＝1	
	B＝0	B＝1	B＝0	B＝1
A & B	0	0	0	1
A|B	0	1	1	1
～A	1	1	0	0
xor(A，B)	0	1	1	0

所有的算法语言中都有逻辑运算。MATLAB 的特点是使逻辑运算用于元素群，得出同阶的 0—1 矩阵。为了按列、按行判断一群元素的逻辑值，它又增加了两种对元素群的逻辑运算函数 all（全为真）和 any（不全为假），见表 2－10。

现在来看逻辑式 u＝p|～p，这是把 p 和“非”p 求“或”。～p 就是把 p 中的 0 元素换成 1，1 元素换成 0。在每个元素位置上，必有一个是 1，把 p 和～p“或”起来，一定是全 1。得

```
u = 1    1    1    1    1    1
    1    1    1    1    1    1
    1    1    1    1    1    1
    1    1    1    1    1    1
    1    1    1    1    1    1
    1    1    1    1    1    1
```

all 和 any 后的输入变量应为矩阵，它是按列运算的。从它们的定义可知

```
all(p)=  0    0    0    0    0    0    (列中有一个元素为 0，即得 0)
all(u)=  1    1    1    1    1    1    (列中元素为全 1，才得 1)
any(p)=  1    1    1    1    1    1    (列中有一个元素为 1，即得 1)
```

表 2－10 运算符和逻辑函数库(ops)(n)

数学及逻辑运算符			
符 号	意 义	符 号	意 义
+	加	—	减
/	矩阵左除或右除	...	行命令延续符
.*	矩阵元素乘	./	矩阵元素除
( ) { }	优先，下标输入参量	[ ]	矩阵，向量输出变量
.	小数点	..	母目录
,	语句分割符，显示	;	语句分割符，不显示
‘	转置，引用	!	操作系统命令
==	关系相等符	< >	关系大小符
&	逻辑与	|	逻辑或
xor	异或	kron	Kronecker 积
*	矩阵乘	=	赋值符
^	矩阵乘幂	%	注释符
.^	矩阵元素乘幂	~=	关系不等符
:	整行(列)等增量赋值	~	逻辑非
逻 辑 函 数			
exist	检查变量或函数是否有定义	any	检查向量中有无非零元素
all	检查向量中元素是否全为非零	find	找到非零元素的序号或下标
isnan	元素为 NaN 时得 1	isinf	元素为 Inf 时得 1
isfinite	元素为有限值时得 1	isempty	矩阵为空阵时得 1
isreal	矩阵为实数阵时得 1	issparse	矩阵为稀疏阵时得 1
isstr	为文本字符串时得 1	isglobal	变量为全局变量时得 1
位 运 算			
bitand *	按位求“与”	bitcmp *	按位求“非”(补)
bitor *	按位求“或”	bitmax *	最大浮点整数
bitxor *	按位求“异或”	bitset *	设置位
bitget *	获取位	bitshift *	按位移动
集 合 运 算			
union *	集合“合”	unique *	去除集合中的重复元素
intersect *	集合“交”	setdiff *	集合“差”
setxor *	集合“异或”	ismember *	是集合中的元素时为真

2.4.3　流程控制语句

计算机程序通常都是从前到后逐条执行的。但有时也会根据实际情况，中途改变执行的次序，称为流程控制。MATLAB 设有四种流程控制的语言结构，即 if 语句、while 语句、for 语句和 switch 语句。

1. if 语句

根据复杂程度，if 语句有三种形式。

(1) if(表达式)语句组 A，end。

其流程如图 2－1(*a*)所示。执行到此语句时，计算机先检验 if 后的逻辑表达式，如为 1，就执行语句组 A；如为 0，就跳过语句组 A，直接执行 end 后的后续语句。注意，这个 end 是决不可少的，没有它，在表达式为 0 时，就找不到继续执行的程序入口。

(2) if(表达式 1) 语句组 A，else 语句组 B，end。

其流程见图 2－1(*b*)。执行到此语句时，计算机先检验 if 后的(逻辑)表达式，如为 1，就执行语句组 A；如为 0，就执行语句组 B。else 用来标志语句组 B 的执行条件，同时也标志语句组 A 的结束(免去了 end)。同样，最后的 end 是不可少的，没有它，执行完语句组 A 后，就找不到进入后续程序的入口。

(3) if(表达式 1) 语句组 A，elseif (表达式 2)语句组 B，else 语句组 C，end。

其流程见图 2－1(*c*)。前两种形式的 if 语句都是两分支的程序结构，要实现两个以上分支的结构就得采用含 elseif 的结构。图中表示的是三分支的情况。在语句中间可加入多个 elseif 以形成多个分支，只是程序结构显得冗长。

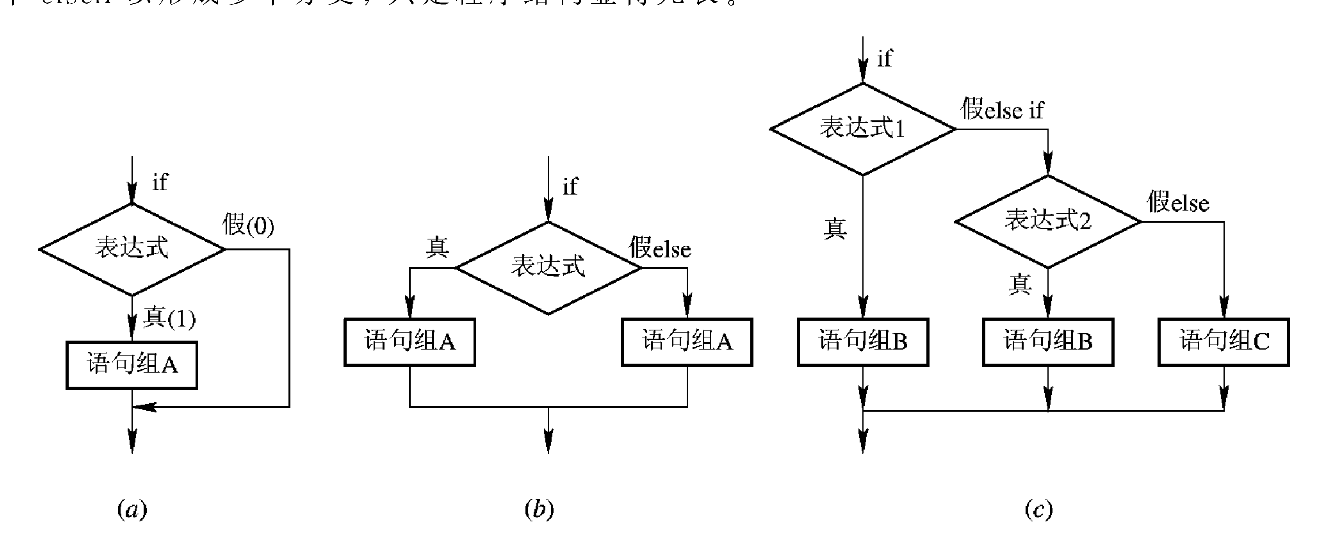


图 2－1　if 语句的三种程序结构形式

【例 2－4－1】　输入数 n，判断其奇偶性。程序如下

n＝input('n＝')，if rem(n，2)＝＝0 A＝'even'，else A＝'odd'，end

运行此程序时，程序要求用户输入一个数，然后判断该数是奇数还是偶数。所以该程序共有两个出口。实际上这个程序并不全面，如果用户根本未键入任何数就回车，程序会判断为“odd”(请读者考虑其原因)。为了使程序在用户无输入时自动中止，可以把程序改为

if isempty(n)==1 A='empty', elseif rem(n, 2)==0 A='even', else A='odd', end

实际上这个程序仍不全面，它不能用于负数，请读者分析其原因。

2. while 语句

while 语句的结构形式为

while (表达式)语句组 A, end

其流程见图 2-2。执行到此语句时，计算机先检验 while 后的逻辑表达式，如为 1，它就执行语句组 A；到 end 处后，它跳回到 while 的入口，再检验表达式；如还是 1，再执行语句组 A；周而复始，直到表达式不成立(结果为零)为止。此时跳过语句组 A，直接执行 end 后的后续语句。与 if 语句的不同在于它的分支中是循环地执行某个语句组，故称为循环语句。

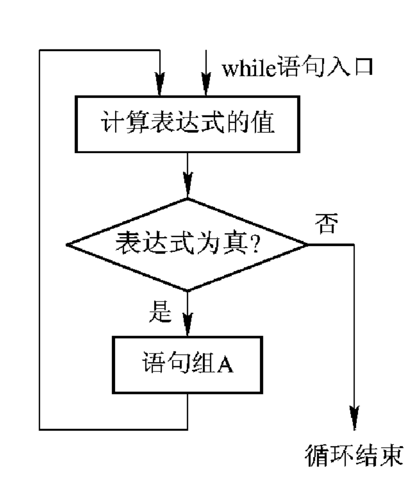


图 2-2 while 语句流程图

【例 2-4-2】 求 MATLAB 中的最大实数。

解：我们设定一个数 x，让它不断增大，直到 MATLAB 无法表示它的值，只能表示为 inf 为止。于是，可列出下列程序

```
x=1; while x~=inf, x1=x; x=2*x; end, x1
```

其中我们先设 x=1，进入 while 循环，只要 x 不等于 inf，就把 x 加倍，直到 x=inf。如果把此时的 x 显示出来，它是无穷大，不是题中要找的数。要找的是变为无穷大之前的最大数，因此在对 x 加倍之前，把它存在 x1 中，显示的 x1 就是要求的最大数。运行这行此程序，得

x1= 8.9885e+307

系统的最大浮点实数为 $(2-\varepsilon)*2^{1023}$(见表 2-1)，其十进制形式为

realmax = 1.7977e+308

两者数量级接近，但还是相差将近一倍，这是因为我们每次把 x 翻一番，故求得的数可能比最大数小不到一半。如果把程序中的 x=2*x 改为 x=1.1*x，结果就会准确一些，得到

x1=1.783718732622142e+308

【例 2-4-3】 求 MATLAB 的相对精度。

解：解的思路是让 y 不断减小，直至 MATLAB 分不出 1+y 与 1 的差别为止。其程序为

```
y=1; while 1+y>1, y1=y; y=y/2; end, y1
```

结果为

y1=2.220446049250313e-016

与 MATLAB 内部给出的浮点数相对精度 2^{-52}(见表 2-1)的十进制数相同。

3. for 语句

for 语句的结构形式为

for k=初值：增量：终值 语句组 A，end

即它把语句组 A 反复执行 N 次。在每次执行时程序中的 k 值不同。有多少个 k 值呢？可得

N=1+(终值-初值)/增量

【例 2-4-4】 用 for 语句求三角函数表的程序为

```
for x=0：0.1：pi/4 disp([x，sin(x)，cos(x)，tan(x)])，end
```

所得的结果将和 2.3 节中的答案相同。这也可以看出，MATLAB 的元素群运算功能与一个 for 循环相当。由于它不需每次检验表达式，因此运算速度比 for 语句快得多。但是不能认为它可全部取代 for 语句，由下例可以看出。

【例 2-4-5】 列出构成 hilbert 矩阵的程序，它需要两重循环：

```
n=input('n=')，format rat
for i=1：n，for j=1：n，h(i，j)=1/(i+j-1)；end，end，h
```

执行时，先按提示输入 n，比如输入 5，结果为

```
h=  1     1/2    1/3    1/4    1/5
    1/2   1/3    1/4    1/5    1/6
    1/3   1/4    1/5    1/6    1/7
    1/4   1/5    1/6    1/7    1/8
    1/5   1/6    1/7    1/8    1/9
```

为了改善可读性，对于流程控制语句，最好用缩进的方法来编写程序。如本例中应写成：

```
format rat，n=input('n=')，
for i=1：n
    for j=1：n
        h(i，j)=1/(i+j-1)；
    end
end
h
```

由于我们现在是在 MATLAB 命令窗中直接输入程序，因此不得不把它写在一行中。此时要注意，在 if，for，while 与表达式之间应留空格，在表达式与语句组之间必须用空格或逗号分隔，而在语句组的后面，必须要用逗号或分号来与 end 或 else 相分隔，否则，MATLAB 会显示出错信息并中止运行。

break 是中止循环的命令，在循环语句中，可用它在一定条件下跳出循环，这是常常用到的。在多重循环中，break 只能使程序跳出包含它的最内部的那个循环。

4. switch 语句

switch-case-otherwise 语句是一种均衡快速的多分支语句，其基本语言结构可表达为：

```
switch 表达式(标量或字符串)
case   值 1
语句组 A
case   值 2
语句组 B
```

```
    …
otherwise
语句组 N
    end
```

当表达式的值(或字符串)与某 case 语句中的值(字符串)相同时，它就执行该 case 语句后的语句组，而后直接跳到终点的 end。case 语句可以有 N－1 个，也即可以有 N－1 个分支，如果没有任何一个 case 值能与表达式值相符，则将执行 otherwise 后面的语句组 N。

例如，判断输入数 n 的奇、偶、空的程序可用 switch 语句写成

```
switch mod(n，2)，case 1，A＝′奇′，case 0，A＝′偶′，otherwise，A＝′空′，end
```

注意，把它写成单行命令时的标点格式，其中有些逗号可以用分号代替，但不得省略。另外为了包含负数中的奇数，将前面例中的 rem 改为 mod，读者可从 rem(－3，2)和 mod(－3，2)的差别得出结论。在正式写程序时，每个 case 语句必须写在行首，以增强程序的可读性。

2.5 基本绘图方法

MATLAB 可以根据给出的数据，用绘图命令在屏幕上画出其图形，通过图形对科学计算进行描述。这是 MATLAB 独有的优于其他语言的特色。在 MATLAB 中可选择多种类型的绘图坐标，可以对图形加标号、加标题或画上网状标线。这些命令属于 graph2d 函数库(见表 2－12)，另外，还有一些命令可用于屏幕控制，坐标比例选取以及在打印机上进行硬拷贝等等，这些命令放在 graphics 子目录中(见表 2－13)。三维及颜色绘图命令放在 graph3d 子目录中(见表 2－14)。还有一些特殊绘图命令放在 specgraph 子目录中(见表 2－15)。我们不可能在此介绍所有的命令，但大部分命令会在本书中涉及，下面分六个问题来讨论。

2.5.1 直角坐标中的二维曲线

plot 命令用来绘制 x－y 坐标中的曲线。它是一个功能很强的命令。输入变量不同，可以产生很多不同的结果。

1. plot(y)——输入一个数组的情况

如果 y 是一个数组，函数 plot(y)给出线性直角坐标的二维图，以 y 中元素的下标作为 X 坐标，y 中元素的值作为 Y 坐标，一一对应画在 X－Y 坐标平面图上，而且将各点以直线相连。例如，要画出十个随机数的曲线。可列出：

```
y＝5＊(rand(1，10)－.5)
y ＝ －1.4052   －2.2648   0.8943   0.8965   2.1735   －0.5825   0.0971   1.6548
 －2.3271   －2.2327
```

由 Rand 函数产生的随机数的最大值为 1，最小数为 0，平均值为 0.5。所以 y 的最大值为 2.5 最小值为－2.5，平均值为 0。键入 plot(y)，MATLAB 会产生一个图形窗，自动规定最合适的坐标比例绘图。X 方向是下标，从 1～10，Y 方向范围则是－4～4，并自动标

出刻度。可以用 title 命令给图加上标题，用 xlabel，ylabel 命令给坐标轴加上说明，用 text 或 gext 命令可在图上任何位置加标注，也可用 grid 命令在图上打上坐标网格线。

```
title('my first plot')
xlabel('x'), ylabel('Y')
grid
```

这时形成的图如图 2－3 所示。

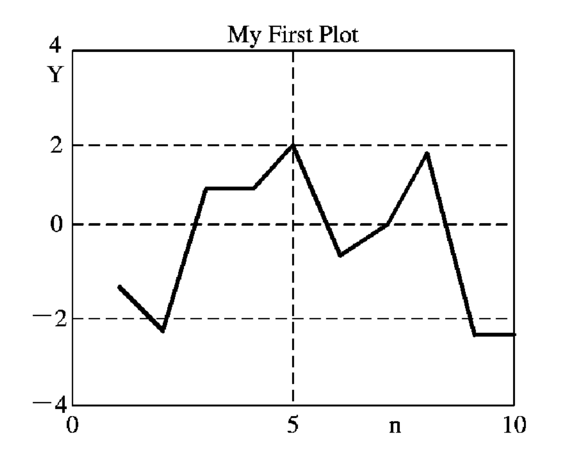


图 2－3　第一张简单的随机数图

2．plot(x，y)——输入两个数组的情况

如果数组 x 和 y 具有相同长度，命令 plot(x，y)将绘出以 x 元素为横坐标，y 元素为纵坐标的曲线。例如，设 t 为时间数组 t=0：0.5；4 * pi，y 是一个随 t 作衰减振荡的变量：y=exp(－0.1 * t). * sin(t)，则 plot(t，y) 就以 t 为横坐标，y 为纵坐标画曲线。如图 2－4 中的实线曲线。若设 y1=exp(－0.1 * t). * sin(t+1)，则由 plot(t，y1，'：')画出的曲线，其正弦波的相位超前了 1 弧度，因此其波形如图 2－4 中的虚曲线所示。实际上，在绘制第二条曲线时，如不加别的命令，第一条曲线就会自动消失。不会有两根曲线同在图中出现。为了在一张图中绘制多条曲线，要用到后面所说的方法。

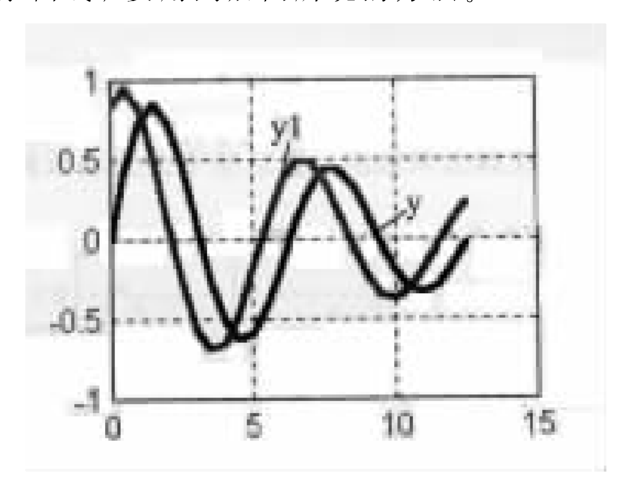


图 2－4　两根曲线画在同一图上

2.5.2　线型、点型和颜色

MATLAB 会自动设定所画曲线的颜色和线型。如果用户对线型的默认值不满意，可以用命令控制线型，也可以根据需要选取不同的数据点的标记。为了设定线型，在输入变量组的后面，加一个引号，在引号内部放入线型和颜色的标志符，如

```
plot(x, y, '*b')
```

这样绘出的图线，其数据点处均用 * 作蓝色标记，而各点之间不再连以直线。

```
plot(x1, y1, ':y'), plot(x2, y2, '+r')
```

绘出的第一条曲线是黄色的点线，第二条曲线的数据点标记为红色的“＋”号。其他线型点型和颜色见表 2－11。

表 2 - 11 线型、点型和颜色

标 志 符	颜 色	标 志 符	线型和点型
y	黄	.	点
m	品红	o	圆圈
c	青	×	×号
r	红	+	+号
g	绿	—	实线
b	兰	*	星号
w	白	:	虚线
k	黑	—.	点划线
		——	长划线

2.5.3 多条曲线的绘制

在一张图上画多根曲线有四种方法。

1. 用 plot(t, [y1, y2, …])命令

该语句中 t 是向量，y=[y1, y2, …] 是矩阵，若 t 是列(行)向量，则 y 的列(行)长与 t 长度相同。y 的行(列)数就是曲线的根数。例如，

plot(t, [y; y1])

就得出图 2 - 4 中的曲线。它会自动给曲线以不同的颜色。这种方法要求所有的输出量有同样的长度和同样的自变量向量。另外它不便于用户自行设定线型和颜色。

2. 用 hold 命令

在画完前一张图后用 hold 命令保持住，再画下一条曲线。如键入

plot(t, y), hold on, plot(t, y1, 'g')

执行此命令时，图形窗产生第一幅图形，同时，命令屏幕显示 Current plot held，图形处于保持状态。再执行 plot(t, y1, 'g')，就把第二幅图以绿色的曲线迭合在同一张图上了。

用这种方法时两张图的变量长度可以各不相同。只要每张图自己的自变量和因变量同长即可。例如，再给一组数据[t2, y2]，其点数比[t, y]多，但占的时间却短。键入

```
t2=0:.2:2 * pi;
y2=exp(-0.5 * t2). * sin(5 * t2+1);
plot(t2, y2)
```

得出的图形应如图 2 - 5 中较短的那条曲线(但线型不同)。用这种方法时，一是注意第一张图的坐标要适当，以保证能看清第二张图；因为用第一种方法时，坐标系是系统自动按多根曲线的数据综合选取的，不会有选择不当的问题；二是注意及时解除保持状态，即键入 hold off，否则，以后的图都会叠加在此图上，造成混乱。

3. 在 plot 后使用多输入变量

在 plot 后使用多输入变量所用语句为

plot(x1, y1, x2, y2, …, xn, yn)

其中 x1，y1，x2，y2 等分别为数组对。每一对 x－y 数组可以绘出一条图线，这样就可以在一张图上画出多条图线，每一组数组对的长度可以不同，在其后面都可加线型标志符。例如，键入

```
plot(t, y, '+g', t2, y2, ': r')
title('线型，点型和颜色')
xlabel('时间'), ylabel('Y')
```

执行这些语句就得到图 2－5。一根图线在数据点处用绿色的＋号作标记，另一根图线用红色。注意我们用的是汉字标注，MATLAB 也照样把汉字标在图上。因为在引号中的内容，MATLAB 只作为一种代码来传递。

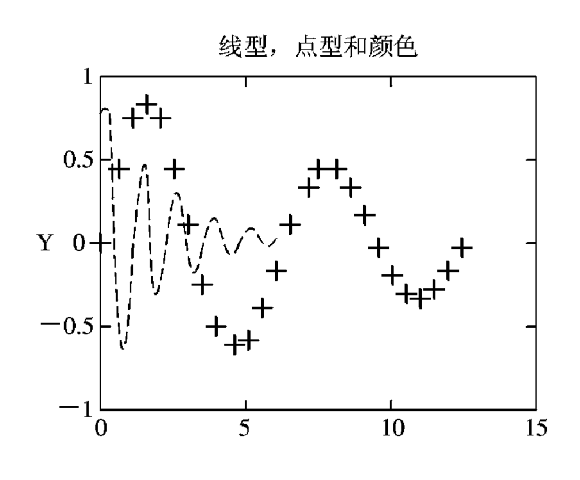


图 2－5　两组长度不同的[t，y]数据画在同一图上

4．用 plotyy 命令

plotyy 设有两个纵坐标，以便绘制两个 y 尺度不同的变量，但 x 仍只能用同一个比例尺，例如，键入

```
y3=5*y2; plotyy(t, y', t2, y3)
```

就得到图 2－6。其中左纵坐标是对 y 的，而右纵坐标是对 y3 的，纵坐标和曲线的标注可用 gtext 命令：

```
grid, gtext('t, t2')
gtext('y'), gtext('y3')
```

gtext 命令用鼠标拖动来确定标注文字的位置，用起来比较方便。

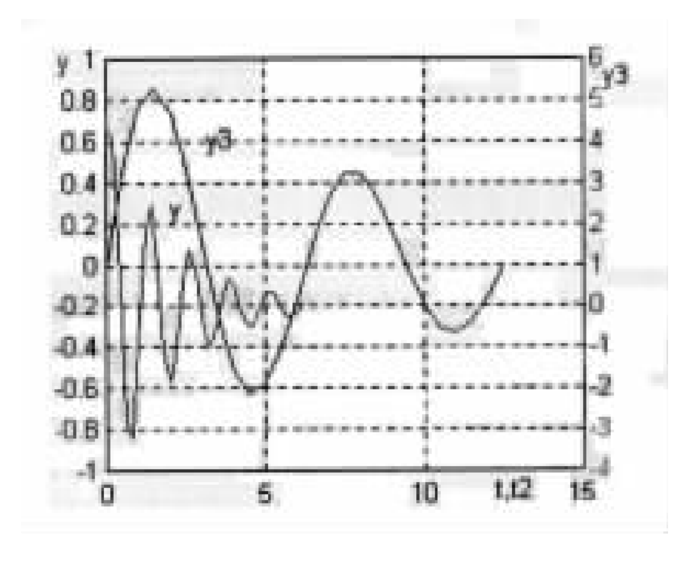


图 2－6　双纵坐标绘图

2.5.4　屏幕控制和其他坐标二维绘图

1．图形屏幕控制命令(见表 2－12)

图形屏幕可以开或关，可以开几处图形窗，也可以在一个图形窗内画出几幅分图，几幅分图也用不同的坐标。以下几种命令可以实现图形窗口间的转换和清除：

(1) figure：打开图形窗口。MATLAB 中的第一幅图随 plot 命令自动打开，以后的

plot 命令都画在同一张图上。如要画在另一张新图上，就要用 figure 命令打开新的图形窗口。有了顺序为 1，2，3，…的几个图形窗后，再用 plot 语句，就要指明画在哪张图上，即键入 figure(i)，表示打开第 i 幅图。否则，所有的图都会画在最后显示的那幅图上。

(2) clf：清除当前图形窗的内容(也可用 clg，但以后将被淘汰)。

(3) hold：保持当前图形窗的内容，再键入 hold，就解除冻结。这种拉线开关式的控制有时会造成混乱，可以用 hold on 和 hold off 命令以得到确定的状态。

(4) close：关闭当前图形窗。close all：关闭所有图形窗。

(5) subplot(n，m，p) 命令：将图形窗口分为 n×m 个子图，在第 p 个子图处绘制图形。

2. 其他二维绘图命令(见表 2－12)

在线性直角坐标系中，绘制其他形式图的命令有 stem(绘脉冲图)、stairs(绘阶梯图)、bar(绘条形图)、errorbar(绘误差条形图)和 hist(绘直方图)等。这些函数用法与 plot 相仿，但没有多输入变量形式。fill(t，y，'颜色标注符')在曲线和坐标轴之间的封闭区填以指定的颜色。

下列程序把画面分成四个：

```
subplot(2, 2, 1), stem(t, y)
title('stem(t, y)'), pause
subplot(2, 2, 2), stairs(t, y)
title('stairs(t, y)'), pause,
subplot(2, 2, 3), bar(t, y)
title('bar(t, y)') , pause
subplot(2, 2, 4), fill(t, y, 'r')
title('fill(t, y, ''r'')')
```

其运行的结果见图 2－7。读者不难从中弄清这几条绘图命令的意义。程序中最后一行，在 r 前后的引号写成两个引号，这是因为它是处在 title 后的引号内。MATLAB 规定，这种引号必须写成两个，以免混淆。

再键入 subplot(1，1，1)命令可取消子图，转回全屏幕绘图。

在对数直角坐标系中的绘图命令有 loglog、semilogx 和 semilogy 等，在极坐标系中的绘图命令有 polar。它们的用法与 plot 的基本用法同。只是数据将画在不同类的坐标系上，可参看表 2－12。

- polar(theta，rho)为极坐标绘图，以角度 theta 为一个坐标，单位是弧度，另外一个是矢径 rho。.在其后使用 grid 命令，可以绘出网状极坐标线。这个函数没有多输入变量形式。
- loglog 绘出以 log10－log10 为坐标刻度的对数图。
- semilogx 使用半对数刻度绘图，x 轴为 log10 刻度，y 轴为线性刻度。
- semilogy 使用半对数刻度绘图，y 轴为 log10 刻度，x 轴为线性刻度。

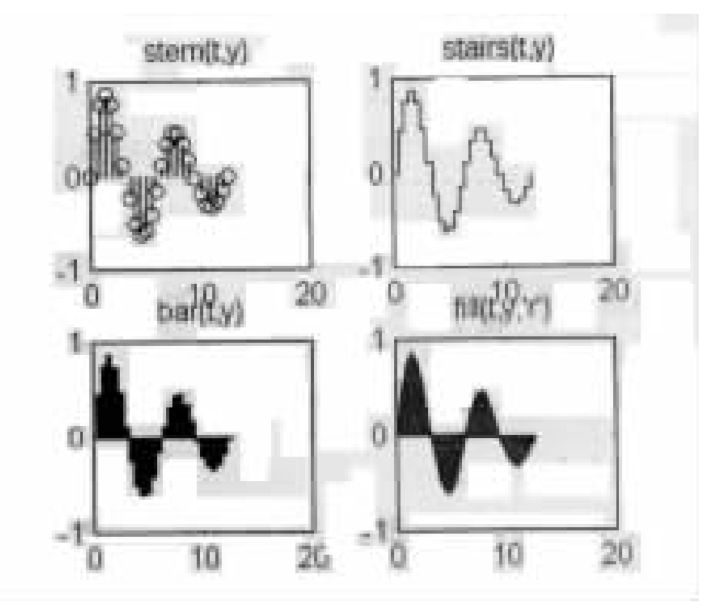


图 2－7　同一函数的几种不同的绘制形式

表 2－12　二维图形函数库(graph2d)(p)

基本 X—Y 图形	plot	线性 X—Y 坐标绘图	polar	极坐标绘图
	loglog	双对数 X—Y 坐标绘图	plotyy	用左、右两种 y 坐标画图
	semilogx	半对数 X 坐标绘图	semilogy	半对数 Y 坐标绘图
坐标控制	axis	控制坐标轴比例和外观	subplot	在平铺位置建立图形轴系
	hold	保持当前图形		
图形注释	title	标出图名(适用于三维图形)	gtext	用鼠标定位文字
	xlabel	X 轴标注(适用于三维图形)	legend	标注图例
	ylabel	Y 轴标注(适用于三维图形)	grid	图上加坐标网格(适用于三维)
	text	在图上标文字(适用于三维)		
打印	print	打印图形或把图存为 M 文件	orient	设定打印纸方向
	printopt	打印机默认选项		

3. 虚数的绘图

当 plot(z)中的 z 为复数单变量时(即含有非零的虚部)，MATLAB 把复数的实部作为 X 坐标，虚部作为 Y 坐标进行绘图。即相当于 plot(real(z)，imag(z))。如果是双变量如 plot(t，z)，则 z 中的虚数部分将被丢弃。要在复平面内绘出多条图线，必须用 hold 命令，或把多根曲线的实部和虚部明确地写出，作为 plot 函数的输入变元，即 plot(real(z1)，imag(z1)，real(z2)，imag(z2))。

例如，要绘制 z＝exp((－.1＋i) * t)的复数图形，列出下面的程序

```
figure(2)
z＝exp((－.1＋i) * t);
subplot(2, 2, 1)
plot(z), pause
```

```
title('复数绘图 plot(z)')
subplot(2, 2, 2)
plot(t, z), pause
title('复数绘图 plot(t, z)')
subplot(2, 2, 3), polar(angle(z), abs(z))
title('polar(angle(z), abs(z))')
subplot(2, 2, 4), semilogx(t, z)
title('semilogx(t, z)')
```

所得图形如图 2-8 所示。其中图 2-8(*a*)画出了复数图形，而图 2-8(*b*)只画了 z 的实部随 t 的变化规律，图 2-8(*c*)是用极坐标绘制的复数曲线，图 2-8(*d*)说明了半对数坐标绘图的结果。

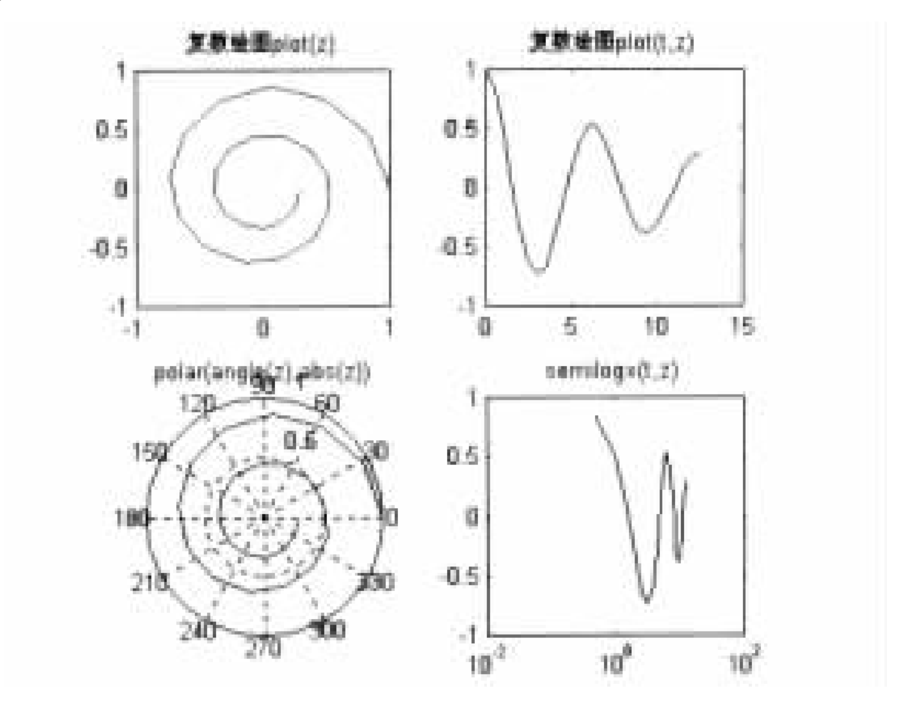


图 2-8　复数绘图及其他坐标轴绘图

4. 坐标比例和尺寸的设定——axis 命令

MATLAB 有根据输入数据自动设定坐标比例的功能。但在有些情况下，用户需要自行设定坐标比例并选择图形边界范围，这时可用 axis 命令。它有多种用法，随输入变量的不同而定。

· V=axis，将返回值当前图形边界的四元行向量，即 V=[xmin, xmax, ymin, ymax]，如果当前图形是三维的，则返回值将是三维坐标边界的六元行向量。

· axis(V)(其中 V 是一个四元向量)，将坐标轴设定在 V 规定的范围内。

· axis 的另外一个功能是控制图形的纵横比。axis('square')或 axis('equal')使屏幕上 x 与 y 的比例尺相同，在这种方式下，斜率为 1 的直线的倾斜角为 45°，对于程序

```
z=0: 0.1: 2*pi; x=sin(z); y=cos(z);
subplot(1, 2, 1), plot(x, y), subplot(1, 2, 2), plot(x, y), axis('equal')
```

虽然数据是圆，但由于屏幕本身长宽不等，因此图 2-8(*a*)得出的是椭圆。图 2-8(*b*)由于函数 axis('equal')的作用，便得出圆(如图 2-9 所示)。axis('mormal')将恢复正常的

纵横尺寸比。键入

　　v＝axis

可得　v＝　－1　1　－1　1

axis 命令的功能非常丰富，这里只介绍了一部分。要想知道详情，可参阅 help axis。

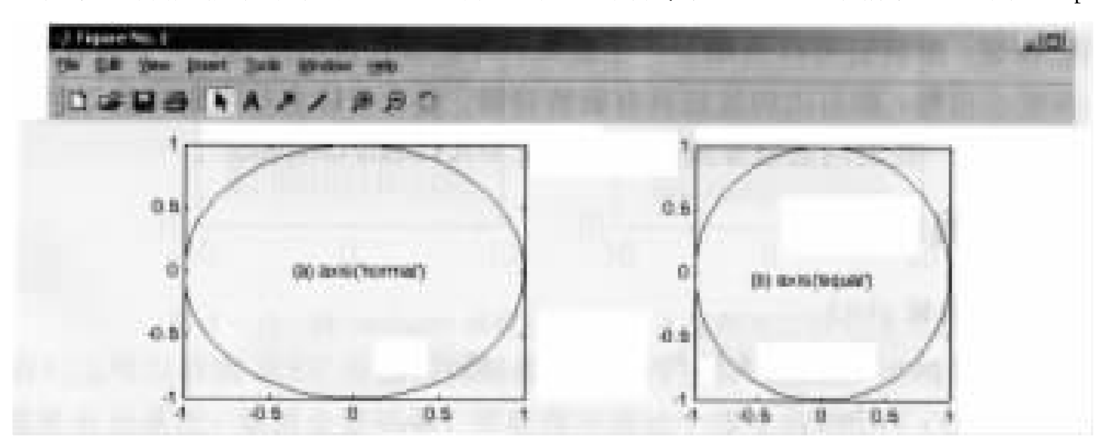


图 2－9　axis('equal')命令的作用

MATLAB 中通用图形函数可参见表 2－13，其中有关图形句柄的命令可先跳过。

表 **2－13**　通用图形函数(**graphics**)(**h**)

图形窗的控制	figure	创建图形窗	shg	显示图形
	gcf	获取当前图形窗的句柄	refresh	刷新图形
	clf	清除当前图形窗	close	关闭图形窗
轴系的控制	axes	在任意位置创建坐标系	ishold	保持当前图形状态为真
	gca	获取当前坐标系的句柄	box *	形成轴系方箱
	cla	清除当前坐标系		
图形对象	line	创建直线	surface	创建曲面
	patch	创建图形填充块	light *	创建照明
	image	创建图像		
图形句柄操作	set	设置对象特性	gcbo	获得回叫对象的句柄
	get	获得对象特性	gcbf	获得回叫图形的句柄
	reset	复位对象特性	drawnow	直接等待图形事件
	delete	删除对象	findobj *	寻找具有特定值的对象
	gco	获得当前对象的句柄	copyobj *	为图形对象及其子项作硬拷贝
工具	closerq	请求关闭图形窗	ishandle *	是图形句柄时为真
	newplot	说明 NextPlot 的 M 文件		
杂项	ginput	从鼠标作图形输入	uiputfile	给出存储文件的对话框
	graymon	设定图形窗灰度监视器	uigetfile	给出询问文件名的对话框
	rbbox	涂抹块	whitebg	设定图形窗背景色
	rotate	围绕指定方向旋转对象	zoom	二维图形的放大和缩小
	terminal	设定图形终端类型	warndlg	警告对话框

5. 图形窗中的直接编辑

除了用行命令来给图形窗设定坐标，添加文字之外，还可以直接用图形窗编辑功能。图 2－9 中除图形外，还把图形窗顶部的菜单和按钮都显示出来了。着重看按钮，从左至

右，前四个按钮不必说，第五个是"选择"按钮，激活它(变白)后鼠标就可在图中选择特定的对象，例如选定整个子图，图中的一根曲线或一组文字。然后可以移动它的位置，也可以点击菜单上的 Edit 项对此对象进行编辑修改；第六个是"文字"按钮，激活它可以输入文字，包括英文、汉字和拉丁字母；第七个是"箭头"按钮，激活它可以产生标注的箭头；第八个是"直线"按钮，激活它可以在图上产生直线；再看靠右边的按钮，一个具有放大功能，另一个具有缩小功能；最右边的按钮具有旋转功能。读者可以自行摸索试验以掌握他们的功能和用法，深入一些的问题需参阅手册"Using MATLAB Graphics"。

2.5.5 三维曲线和曲面

1. 空间曲线绘制 plot3

空间曲线绘制 plot3 的用法大体与 plot 相仿。格式为

plot3(x, y, z, 's')。

其中，s 为线型颜色符。例如

```
z=0: 0.1: 4 * pi;
x=cos(z);
y=sin(z);
plot3(x, y, z)
```

得出的将是一条空间螺旋线，如图 2-10 所示。

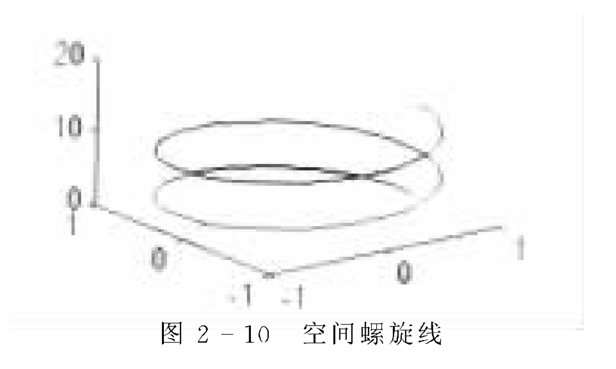


图 2-10 空间螺旋线

2. 空间曲面的绘制

函数 mesh 和 surf 用来绘制三维曲面。三维曲面方程应有 x，y 两个自变量，因此先在 x—y 平面上建立网格坐标，每一个网格点上的数据 z 坐标就定义了曲面上的点。通过直线(mesh)或小平面(surf)连接相邻的点就构成了三维曲面。

mesh 函数可以把一个大矩阵形象化地表示出来。例如，函数 sinc(r)=sin(r)/r (其中 r 是 x—y 平面上的向径)的立体图形是很生动的。可用下面的方法来绘制，程序如下：

```
x=-8: 0.5: 8; y=x';                        % 生成一维的自变量数组
X=ones(size(y)) * x; Y=y * ones(size(x));  % 生成二维的自变量平面
R=sqrt(X. * Y+Y. * Y); z=sin(R)./R;        % 生成因变量
mesh(z), pause                             % 画三维曲面
```

第一行命令定义了函数计算的 x,y 取值范围，每一个方向有 33 个样本点，第二行命令建立了共有 33×33=1089 个网格点的坐标矩阵 X 和 Y，形成了 33×33 网格的矩阵网格，第三行程序表示数据点到原点的距离。并求得 sinc 函数值，最后用 mesh 函数绘出图形。

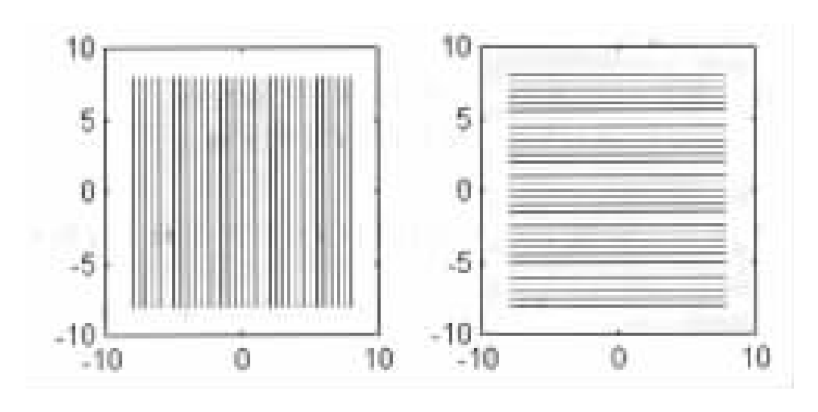
图 2-11 画出 X 是各点的 x 坐标，即竖线，画出 Y 是各点的 y 坐标，即横线。在 z 的定义域内列出 X，Y 后，即可进行函数的计算和绘图。MATLAB 也提供了生成网格点坐标的专用函数 meshgrid。用这个函数时，上述的两行程序可简化为一条语句

```
[X, Y]=meshgrid(-8: 0.5: 8, -8: 0.5: 8);
```

执行了第三行程序后将得出以下的警告

Warning: Divide by zero

即出现了被零除的运算。实际上这发生在 R=0(即原点)处，该处 sin(R)也为零，所以得到



2－11　由 meshgrid 函数求出的 X，Y 矩阵对应的 x，y 坐标

NaN。产生的三维曲线如图 2－12 所示，在 R＝0 处缺掉一个点，因为 NaN 是无法画出的。从这里我们也可以看出 IEEE 算法的优越性。如果按过去的方法，出现零作除数这种非法运算，就要退出系统，取消全部结果，那就得不到这个漂亮的图形了，其实非法运算只是 1089 中点中的一个。不能“攻其一点，不及其余”。解决这个问题的方法也很简单，只要把样本点移离原点即可。为此把程序改为

R＝sqrt(X. ＊ Y＋Y. ＊ Y)＋eps；z＝sin(R). /R；figure(1)，mesh(z)

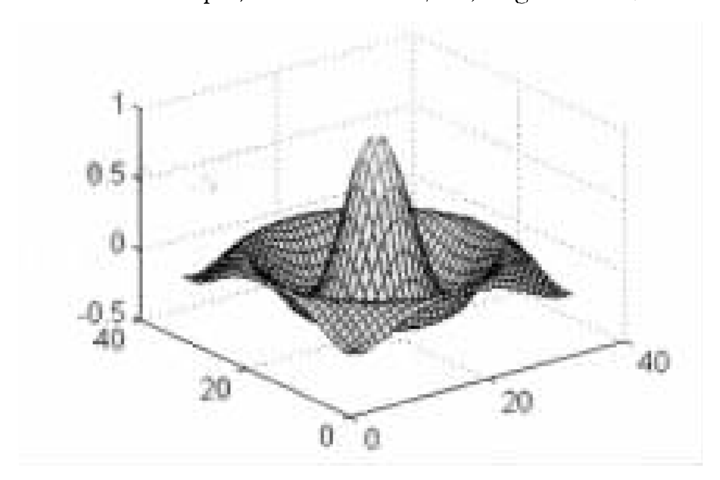


图 2－12　z＝sin(R). /R 的三维曲面图

即把原来的 R 值移动一个极小的数值 eps，运行就没有问题，而图上不再有缺掉的点了。把上式中的 R 改成 abs(x)＋abs(y)(称为一范数，sqrt(x ＊ x＋y ＊ y)称为二范数)，即

R＝abs(X)＋abs(Y)＋eps；z1＝sin(R). /R；figure(2)，surf(z1)

得出的三维曲面如图 2－13 所示。

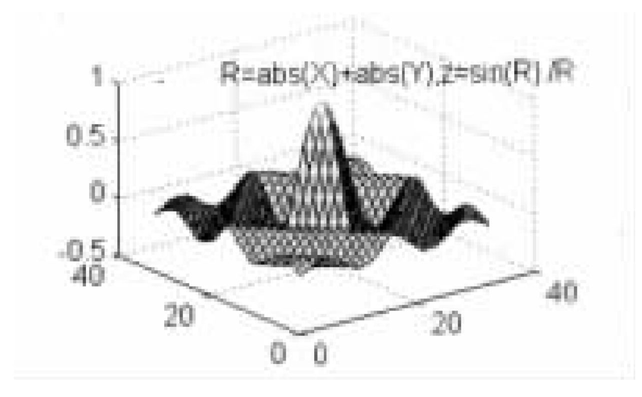


图 2－13　当 R 取一范数时 z＝sin(R). /R 的三维曲面图

3. 其他三维图形绘制命令

三维图形绘制库见表 2－15。其中有一组命令属于外观变换。

· view(方位角,俯仰角)可以变换立体图的视角,例如键入

view(20,0)

就得到另一种三维图形,如图 2－14 中的图(*d*)。(方位角,俯仰角的默认值为[37,30]度。)

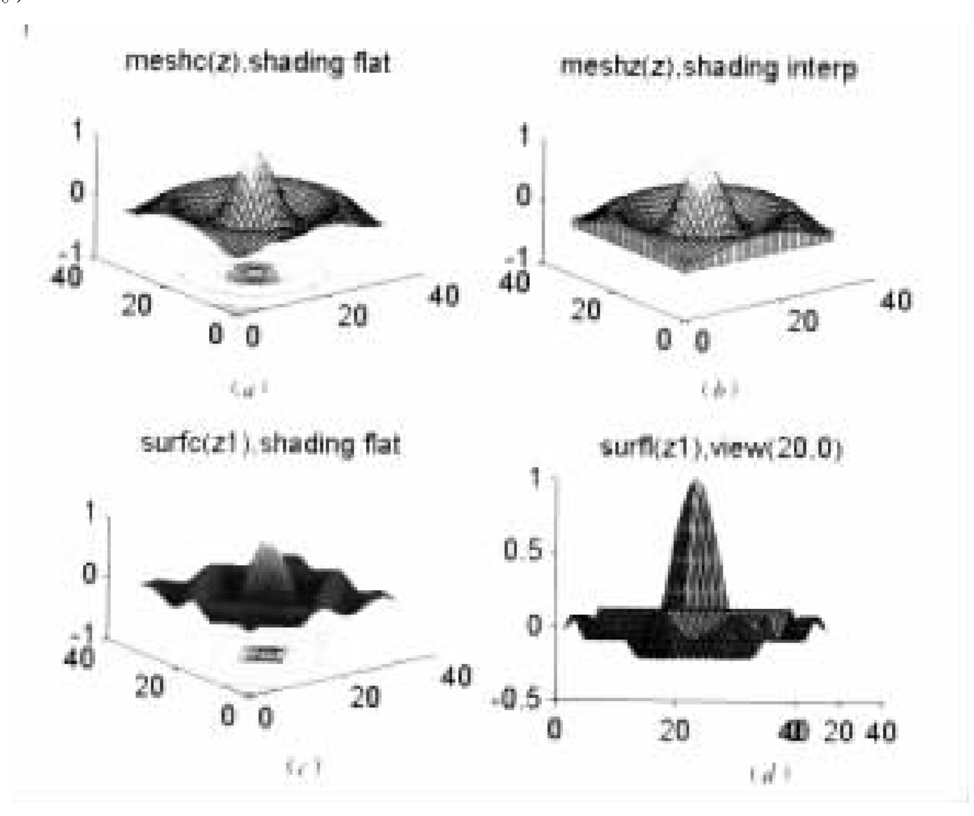


图 2－14 mesh 和 surf 命令的其他形式

· shading flat 或 shading interp 可把曲面上的小格平滑掉,使曲面成为光滑表面。Shading faceted 是默认状态,它使曲面上有小格。MATLAB5 新增了 rotate3d 命令,执行此命令后,用户可以用鼠标拖动立体图形作空间连续转动。

另有一组等高线绘制命令,contour 命令把曲面的等高线投影在 x－y 平面上,也就是普通地图中的画法。contour3 在三维立体图中画出等高线,这些等高线就像云那样浮在相应的高度上。下列程序使用了三维绘图库中的一些命令,其结果可从图 2－14 中看出。有关彩色的命令在后面讨论。

```
subplot(2, 2, 1), R=sqrt(X.^2+Y. * Y); z=sin(R)./R; meshc(z), pause
title('meshc(z), shading flat'), shading flat
subplot(2, 2, 2), R=sqrt(X.^2+Y. * Y)+eps; z=sin(R)./R; meshz(z), pause
title('meshz(z), shading interp'), shading interp
subplot(2, 2, 3), R=abs(X)+abs(Y)+eps; z1=sin(R)./R; surfc(z1), pause
title('surfc(z1), shading flat'), shading flat, % colormap(gray)
subplot(2, 2, 4); surfl(z1), view(20, 0)
title('surfl(z1), view(20, 0)')
```

2.5.6 特殊图形和动画

表 2－14 列出了 MATLAB 中的一些特殊图形函数，其中一部分是各种不同学科和领域中用到的特殊二维和三维图形，例如前面提到过的 stem，stairs 等，pie 和 bar 是在管理科学中常用的饼图和条形图，compass 是电路中常用的相量图，在应用篇中还会介绍，读者可以自行试用。另一部分是前面已介绍过的等高线图形，此处不多占篇幅。

表 2－14 特殊图形库(specgraph)(u)

特殊的二维图形	area	填满绘图区域	feather	羽状图
	bar	条形图	fill	填满两维多边形
	barh	水平条形图	pareto	Pareto 图
	bar3	三维条形图	pie	饼图
	bar3h	三维水平条形图	plotmatrix	矩阵散布图
	compass	极坐标向量图	ribbon	画成三维中的色带
	comet	彗星轨迹图	stem	离散序列绘图
	errorbar	误差条图	stairs	阶梯图
等高线图形	contour	等高线图	pcolor	伪彩色图
	contourf	填充的等高线图	quiver	箭头图
	contour3	三维等高线图	voronoi	Voronoi 图
	clabel	等高线图标出字符		
特殊三维图形	comet3	三维彗星轨迹图	slice	实体切片图
	meshc	三维曲面与等高线组合图	surfc	三维曲面与等高线组合图
	meshz	带帘的三维曲面	trisurf	三角表面图
	pie3	三维饼图	trimesh	三角网状表面图
	stem3	三维 stem 图	waterfall	瀑布图
	quiver3	三维 quiver 图		
图像显示	image	显示图像	imread	从图形文件读出图像
	imagesc	缩放数据并作为图像显示	imwrite	把图像写入图形文件
	colormap	颜色查找表	imfinfo	关于图形文件的信息
电影和动画	capture	从屏幕抓取图形文件	rotate	绕给定方向旋转对象
	moviein	初始化电影帧存储器	frame2im	把电影帧转换为索引图像
	getframe	获取电影帧电影帧	im2frame	把索引图像转换为电影帧
	movie	重放录下的电影帧		
实体	cylinder	生成圆柱体	sphere	生成球体

在这里需要着重介绍的是 MATLAB 的动画命令，它总共有三条命令：moviein、getframe 和 movie。用 getframe 把 MATLAB 产生的图形存储下来，每个图形成一个很大的列向量；再用 N 行这样的列保存 N 幅图，成为一个大矩阵；最后用 movie 命令把它们连起来重放，就可以产生动画效果了。moviein 用来预留存储空间以加快运行的速度(不用也可以)。下面是 MATLAB 手册上提供的一个漂亮的动画程序：

```
axis equal,                     % 因为产生的图形是圆形，故把坐标设成相等比例
M = moviein(16);                % 为变量 M 预留 16 幅图的存储空间
for j=1:16                      % 作 16 次循环
    plot(fft(eye(j+16)));       % 画图
    M(:,j)=getframe;            % 依次存入 M 中
end
```

运行完这几句程序后，16 幅画面(每幅用 16 858 * 8＝134 864 个字节)就放在矩阵 M 中；再键入命令

```
movie(M,30)
```

MATLAB 就以每秒 30 帧的速度播放 M 中的图形。读者可自行在计算机上实践。

2.5.7　彩色、光照和图像

为了更好地显示图形，特别是空间图形，MATLAB 使用了彩色和光照。颜色提供了三维图形中的第四维坐标，扩展了图形表达的能力。光照进一步改善了视觉效果，也是 MATLAB 的一个重要特色。但因为彩色印刷的成本太高，在本书中不便展开，建议读者自己在计算机上实践。

在三维曲面绘图命令中，加入第四维变元，例如 mesh(x，y，z，w)，则 w 的大小就用颜色表示，即在坐标值为(x，y，z)的点上，赋予对应于 w 值的颜色。颜色与 w 值的一一对应关系，用彩色条(colorbar)来表示，用命令 colorbar 可得到这个关系，它将在已有的图形右侧，加上一条垂直的彩色条，并标以对应的 w 值。

如果在三维曲面绘图命令中，没有第四维变元，则颜色轴将与 z 轴一致，有一一对应的关系。例如，键入

```
[x,y]=meshgrid([-2:.2:2]);
z = x.*exp(-x.^2-y.^2);
surf(x,y,z),colorbar
```

所得图形及彩色条如图 2－15(*a*)所示，可见 z 最大处为深红色，z 最小处为深蓝色。

如果把末行改成

```
surf(x,y,z,gradient(z)),colorbar
```

则彩色轴将表示曲面的梯度，也就是产生第四维的坐标，如图 2－15(*b*)所示，可以看出，在峰点谷点之间的斜率最大处为最深的颜色，负斜率处为深蓝色。

彩色条分为 64 个等级，其颜色排序的默认值为 hsv，但可由用户自行修改设定为 hot、cool、gray、copper、pink、jet 和 prism 等。其语句为 colormap(cool)。大家知道，任何颜色均可用红绿蓝三色的适当组合来产生。因此，有左端变量的语句 m＝colormap 将返回一个 64×3 的矩阵。其各列依次表示红、绿、蓝三种基色的分量。rgbplot(m)则用红、绿、蓝三

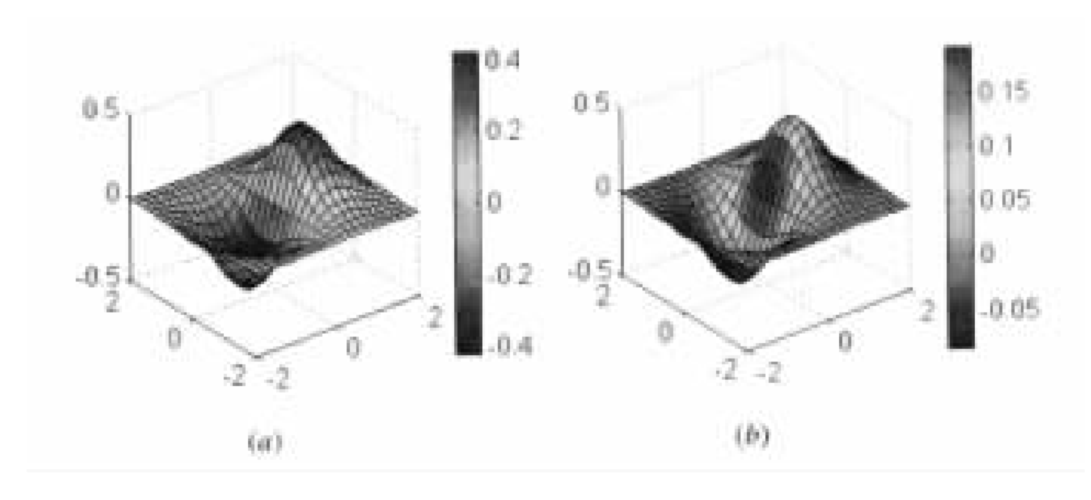


图 2－15　颜色坐标分别表示第三维(z)和第四维(梯度)

种曲线表示三基色分量的大小。表示颜色的另一种方法是色调—饱和度—亮值(hsv)方式，彩色电视机的调色就用这种方式。MATLAB 也提供了两者之间的转换函数。例如对默认的彩色图 m 求 h=rgb2hsv(m)，则 h 也是一个 64×3 的矩阵，只不过它用的是 hsv 颜色表示方式，它的第二、三列全为 1，说明其饱和度及亮值均为 1，仅仅色调在从 0～1 单调地变化，所以它才得到了 HSV 彩色图的名称。这个名称不太好，很容易与 hsv 的颜色表示方式相混淆。

把彩色条离散化形成伪彩色板，它默认地分为 64 份，键入 pcolor([1：65；1：65]′)(65 个分割点产生 64 根彩条)，可得出伪彩色板图，如图 2－16 所示。键入 hot(8)把伪彩色板设置为 hot 色并将其改为 8 份，显示这个伪彩色板应当用命令 pcolor([1：9；1：9]′)。在自动坐标设定的情况下，彩色条应恰好覆盖图中的最小到最大的坐标值(例如在代表 z 值时为[z_{min}，z_{max}])并略有余量。人工设定的命令为 caxis([cmin，cmax])，其作用是把彩色条中的最小最大值分别与 cmin，cmax 相对应，以达到用户特定的显示需要。

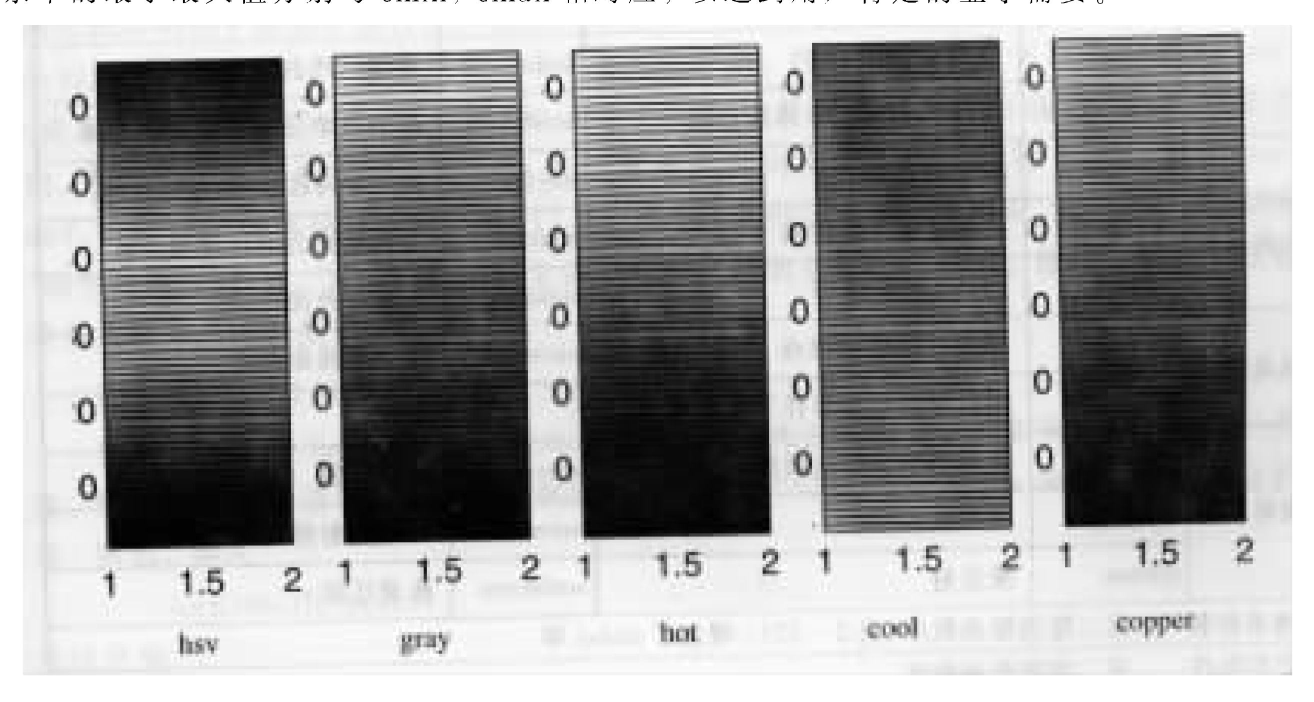


图 2－16　不同彩色设置下的彩色图

在 MATLAB 4 中专门设有颜色和照明函数库，在 MATLAB 5 及以后的版本中，它被并入三维图形库中，见表 2－15。其中光照模型诸函数的输入变元中都要规定光源的方向或坐标，例如 surfl(peaks，[30，0])可得出光源在方位角 30°，俯仰角 0°方向照射时的 peak 曲面。读者可根据应用的需要自行试验其用法，从 help 文本中也可以获得相应的应用信息。

图像和图形的不同在于：图形只有黑白两色，而图像则有深浅之别，称为灰度，彩色图像还有色度。通常黑白图像有 256 个灰度等级，因此图上的每个点要用八位二进制表示。如果 A 是一个 M×N 的矩阵，其每个元素都是八位二进制的整数(0～255)，则 image(A)就创建了一幅 M×N 元素组成的图像，其每个元素按其值在彩色图(colormap)中取色，键入 image(A)，colormap(gray)将得出黑白图像。

表 **2－15** 三维绘图和光照函数库(**graph3d**)(**q**)

绘制三维曲线命令	plot3	在三维空间中画线和点	mesh	三维网格图
	fill3	在三维空间中绘制填充多边形	surf	三维曲面图
颜色控制	colormap	彩色查寻表	caxis	伪彩色坐标轴定标
	shading	彩色阴影方式	hidden	消隐或显示被遮挡的线条
	brighten	改变彩色图的亮度		
彩色图	hsv	色调～饱和度～亮值彩色图	gray	线性灰度彩色图
	hot	黑、红、黄、白彩色图	cool	蓝绿，和洋红阴影彩色图
	bone	蓝色色调的灰度彩色图	copper	铜色调的线性彩色图
	pink	线性粉红色阴影彩色图	prism	光谱彩色图
	jet	HSV 彩色图的变型	flag	红、白、蓝、黑交互的彩色图
	spring	品红和黄阴影彩色图	summer	绿和黄阴影彩色图
	autumn	红和黄阴影彩色图	winter	蓝和绿阴影彩色图
	white	全白彩色图	lines	带颜色线的彩色图
	colorcube *	增强的彩色立方体彩色图	colstyle	从字符串分解出颜色和字体
彩色图有关的函数	colorbar	显示彩色条	hsv2rgb	由 hsv 向红绿蓝(rgb)转换
	rgb2hsv	红绿蓝向 hsv 转换	contrast	变灰度图为对比增强方式
	rgbplot	用 rgb 绘彩色图	spinmap	旋转彩色图
视点控制	view	规定三维图的视点	viewmtx	视点变换矩阵
	rotate3d *	用鼠标拖动图形作三维旋转		
照明模型	surfl	带照明的 3 维曲面图	specular	镜面反射
	lighting	光照模式	material *	材料反射模式
	diffuse	漫反射	surfnorm	曲面法线
轴系控制	见二维图形函数库(表 2－12)，增加了 zlabel 等			
图形标注	见二维图形函数库			
打印输出	见二维图形函数库			

2.5.8　低层图形屏幕控制功能

直到现在，我们所学的都是高层绘图命令。绘图本来是一件很复杂繁琐的工作，MATLAB 怎么把它简化的呢？主要是它代用户做了大量的事务性工作。有些简化是有根据的，例如坐标范围根据输入数据的最大最小值来选择；有许多则没有一定的道理，只是开发人员任意给出的默认值。例如图形的背景色、绘图线条的线宽、线型、颜色、坐标网格的密度及其标注尺寸等。在入门的时候，这些内容用户并不关心，最好由软件开发者帮助设定好，免得分散用户的注意力。但一旦遇到默认值不满足实际应用中的特殊需要时，就会令人觉得高层命令过于自动化和智能化，不便于人的干预，低层图形屏幕控制功能就是为补充这些不足而设定的。

MATLAB 中的图形对象共有 10 类，如图 2－17 所示。

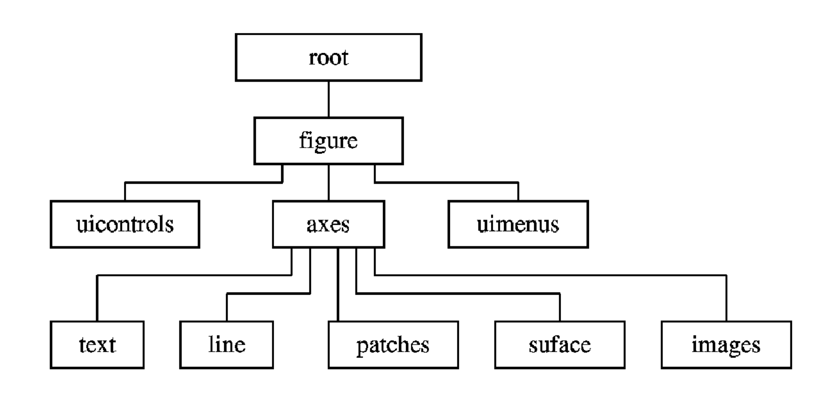


图 2－17　MATLAB 中图形对象的分类和层次关系

root(根对象)在最上层，其子类只有一种 figure(图形对象)。根对象有多个图形对象，在每个图形对象下，又有三个子类，即 uicontrola，uimenus 和 axes。其中，前两类是生成专用人机界面的，这里不加讨论，我们只考虑 axes(坐标系对象)。一个图形对象中可有多个坐标系（如 subplot），在坐标系之下，有六个子类，即 text(图中文字)，line(线)，patches(块)，surface(曲面)、light(光照)和 image(图像)。任何图形都是由这九类图形对象组成的。figure 是 axes 的父类，而 axes 又是其他六种对象的父类。执行 figure，plot，mesh，surf，fill，text 等高层命令时，MATLAB 将生成某些图形对象，同时，也生成了一个称为“图形句柄”的数组，其中包含了产生各个对象所需的各种默认参数。修改其中的参数就可改变当前对象中的图形特性。

1. 取得图形句柄的方法

高层图形命令 figure，plot，mesh，surf 等都是无左端变量的。如果在其左端放一个变量，则执行该绘图语句后，该变量就成为此图形的“句柄”名。用 get 命令可以得到它的内容。例如，键入

```
surh=surfc(pcaks);
```

其结果为

```
surh=   88.0001
        93.0001
          …
```

100.0001

这表明 surfc 命令产生了九个对象。因为 surfc 除产生三维曲面外还产生了八根等高线。其编号分别为 88.0001，93.0001，94.0001，95.0001，96.0001，97.0001，98.0001，99.0001，100.0001，这些编号只有内部的意义，对用户来说，主要记住句柄名，把它们命名为 surh(1)，…，surh(9)。每一个对象都有自己的句柄。例如，键入

get(surh(1))

得

CData=[(49 by 49)]	注：颜色数据
EdgeColor = flat	边缘颜色
EraseMode = normal	可擦模式
FaceColor = [0 0 0]	前景颜色
LineStyle = —	线型
LineWidth = [0.5]	线宽
MarkerSize = [6]	标志点尺寸
MeshStyle = both	网格型式
XData = [(1 by 49)]	X 数据
YData = [(49 by 1)]	Y 数据
ZData = [(49 by 49)]	Z 数据
…	
Children = [ ]	子类对象的句柄
Parent = [56.0001]	父类句柄编号
Type = surface	对象类型

这里我们删略了某些无关的句柄内容。用 gca 命令可得到当前 axes(即上述对象的父类)句柄，用 gcf 命令可得到当前 figure(即上述对象的祖类)句柄，它们的内容更多，此处从略。

2. 修改句柄内容的方法(set 命令)

set(surh(1)，'linewidth'，2，'markersize'，3)

该命令把三维曲面中的线宽加粗到 2 mm，标志点直径减小为 3 mm。但图中的等高线则不受影响。

3. 修改厂设默认参数的方法

前面介绍的方法，只能一张一张图，一根一根曲线地去修改，如果希望把每张图上的曲线都改粗一些，最好一进 MATLAB 环境就把公司出厂设定的默认值改掉。用下述命令可以做到

set(0，'DefaultLineLineWidth'，1.5)

0 代表根对象，把它改了，则它所有的子类和孙类的曲线线宽也都改了。如果我们只需把当前图中的全部线型改变为点划线，可用

set(gcf，'DefaultLineLineStyle'，'—.')

可更改的厂设参数很多，如 Defaultfigruecolor，Defaultsurfaceedgecolor，DefaultAxesColor，DefaultAxesColorOrder，DefaultAxesLineStyleOrder，DefaultTextRotation 等，此处

无法穷尽，读者可参阅 MATLAB 有关手册。

2.6 M 文件及程序调试

在入门阶段，通常让 MATLAB 工作在行命令模式下，键入一行命令后，系统立即执行该命令，用这种方法时，程序可读性很差且难以存储。解决复杂的问题应该用命令编成可存储的程序文本，再让 MATLAB 执行该程序文件，这种工作模式称为程序文件模式。

由 MATLAB 语句构成的程序文件用户可作 M 文件，它将 m 作为文件的扩展名。例如，文件 expml. m 用来计算矩阵指数函数的值。因为它是 ASCII 文本文件，所以可以直接阅读并用任何编辑器来建立。

M 文件可分为两种：一种是主程序，也称为主程序文件(Script File)，是由用户为解决特定的问题而编制的；另一种是子程序，也称为函数文件(Function File)，它必须由其他 M 文件来调用；函数文件往往具有一定的通用性，并且可以进行递归调用(即自己可以调用自己)。MATLAB 的基础部分中已有了近 800 个函数文件，它的工具箱中还有千余个函数文件，并在不断扩充积累。MATLAB 软件的大部分功能都是来自其建立函数集，利用这些函数可以使用户方便地解决他们的特定问题。

2.6.1 主程序文件

主程序文件的格式特征如下：

(1) 用 clear，close all 等语句开始，清除掉工作空间中原有的变量和图形，以避免其他已执行的程序残留数据对本程序的影响。

前几行通常是对此程序用途的说明，特别是在运行时对用户输入数据的要求，更要叙述清楚。不然别人就看不懂也用不成，连自己日后也会遗忘。这些注释行必须以“%”开始，以便计算机执行时不予理会。MATLAB 规定，在键入 help [文件名]时，屏幕上会将该文件中以“%”起头的最前面各行内容显示出来，使用户知道如何使用。这些注释是可以用汉字的。“%”也可以放在程序行的后面，MATLAB 将不执行该字符后的任何内容。

(2) 以下是程序的主体。如果文件中有全局变量，即在子程序中与主程序共用的变量，应在程序的起始部分注明。其语句是

Global 变量名 1 变量名 2 …

为了改善可读性，要注意流程控制语句的缩进及与 end 的对应关系。另外，程序中必须都用半角英文字母和符号(只有引号括出的和%后的内容可用汉字)。特别要注意英文和汉字的有些标点符号(如句号、冒号、逗号、分号、引号乃至%、=、()等)，看起来很相似，其实代码不同。用错了，不但程序执行不通，而且几乎必定死机。因此键入程序时，最好从头到尾用英文，不要插入汉字。汉字可在程序调试完毕后加入。用 MATLAB 5 的编辑器比较好，因为它对出现的非法字符会显示出特殊的颜色，引起用户的注意。并且在它的菜单 View 中，选 Auto Indent Selection 项可以自动对程序进行缩进排版。

(3) 整个程序应按 MATLAB 标识符的要求起文件名，并加上后缀 m。用 MATLAB 4 时，文件名长度不要超过八个字符。不允许用汉字，因为这个文件名也就是 MATLAB 的

调用命令，它是不认汉字的。将文件存入自己确定的子目录中，该子目录应置于 MATLAB 的搜索路径下，所以应避免出现汉字路径名。

完成程序编制，并在 MATLAB 的命令窗中键入此程序的文件名后，系统就开始执行文件中的程序，主程序文件中的语句将对工作空间中的所有数据进行运算操作。

【例 2－6－1】 要求列写一个求 Fibonnaci 数的程序。它是一个数列，从[1，1]开始，由数列的最后两个元素之和生成新的元素，依次递推。其程序如下：

```
% 计算 Fibonnaci 数的 M 文件
clear, close all
N=input('输入最大数值范围 N= ')
f=[1, 1]; i = 1;              % 变量的初始化
while f(i)+f(i+1)<N          % 循环条件检验
    f(i+2)=f(i+1)+f(i); i=i+1;        % 求 Fibonnaci 数的算式
end,
f, plot(f)              % 显示和绘图
```

将此程序以文件名 fibon.m 存入某一 MATLAB 搜索目录下。在 MATLAB 命令窗中键入 fibon，系统就开始执行这个程序，它首先会要求用户输入 N，然后计算数值小于 N 的 Fibonnaci 数，并绘出图线。设输入 N＝100，得出(图线略)：

```
f=1    1    2    3    5    8    13    21    34    55    89
```

该文件执行结束，变量 f 和 i 仍保存在工作空间中。

【例 2－6－2】 列出求素数的程序。所谓素数就是只能被它自身和 1 除净的数。程序如下：

```
% 求素数(prime number)的程序
clear, close all
N =input('N= '), x=2:N;                    % 列出从 2 到 N 的全部自然数
for u=2:sqrt(N)                            % 依次列出除数(最大到 N 的平方根)
    n=find(rem(x, u)==0 & x~=u);           % 找到能被 u 除净而不等于 u 的数序号
x(n)=[];                                   % 剔除该数
end, x                                     % 循环结束，显示结果
```

以 prime.m 为名存入系统，就可以执行了。给出 N＝40，结果为

```
x=  2    3    5    7    11    13    17    19    23    29    31    37
```

2.6.2 人机交互命令

在执行主程序文件中，往往还希望在适当的地方对程序的运行进行观察或干预，这时就需要人机交互的命令，调试程序时，人机交互命令更不可少。下面介绍几条人机交互命令，见表 2－16。

· echo on(off)：一般情况下，M 文件中的命令不会显示在屏幕上。而在命令 echo on 之后，会在执行每行程序前先显示其内容。

· pause(n)：程序执行到此处，暂停 n 秒，再继续执行。如果没有括号参数，则等待用户键入任意键后才继续执行。

· keyboard：程序执行到此处暂停，在屏幕上显示字符 K，并把程序的输入和执行权交给用户（键盘）。用户可以像在普通 MATLAB 命令窗中那样进行任何操作（例如检查中间结果等）。如果需要系统恢复运行原来的程序，只需键入字符串 return。在 M 文件中设置该命令，有利于进行程序调试以及临时修改变量内容。

· input('提示符')：程序执行到此处暂停，在屏幕上显示引号中的字符串。要求用户输入数据。如程序为 X=input('X=')，则屏幕上显示 X=，输入的数据将赋值给 X。数据输入后，程序继续运行。input 命令也可以接受字符，其格式为 Y=input('提示符','s')，此时 Y 将等于输入的字符串。

· ^C：强行停止程序运行的命令。^C 念作(Control-C)，即先按下 Ctrl 键，不抬起再按 C 键。在发现程序运行有错或运行时间太长时，可用此方法中途终止它。

· menu：是用来产生人机交互备选择菜单的命令，参阅 help 文件。

· uimenu，uicontrol：这些是用来产生人机交互面板按钮的命令，参阅 help 文件。

后面两条命令是为编制较复杂的程序的图形界面时使用的，无法用三言两语说明。读者可在用到时再去查阅有关资料。

表 **2 - 16** 语言结构库(**lang**)(**k**)

	名 称	功 能	名 称	功 能
估值并执行	eval	执行 MATLAB 语句字符串	feval	执行由字符串命名的函数
	evalin	估值工作空间中的表示式	builtin	从超载方法执行内置函数
	assignin	分配工作空间中的变量	run	运行程序文件
流程控制语句	if	条件执行命令	else	与 if 联用
	elseif	与 if 联用	end	for，while，if 语句的终点
	for	确定次数的重复语句	while	非确定次数的重复语句
	break	终止执行循环	return	返回到调用函数
	switch	在表示式的几种情况中选择	otherwise	switch 语句中的默认值
	case	switch 语句中的情况		
程序、函数和变量	script	MATLAB 程序文件－M 文件	function	加入新函数
	global	定义全局变量	mfilename	当前执行的文件名
	list	以逗号分割的清单	isglobal	是全局变量时为真
	exit	检查变量或函数是否存在		
变元管理	nargchk	检验输入变元的数目	nargin	输入变元的数目
	nargout	输出变元的数目	varargin	长度可变的输入变元清单
	varargout	长度可变的输出变元清单	inputname	输入变元的名称
信息显示	error	跳出函数并显示信息	lasterr	最近的出错信息
	warning	显示警告信息	errortrap	在测试中跳过错误
	disp	显示数组	fprintf	显示格式化信息
	sprintf	把格式化数据写成字符串	echo	显示执行的 MATLAB 语句
人机交互命令	input	提示用户输入	keyboard	调用等待键盘输入
	menu	生成用户输入的选择菜单	pause	暂停，等待用户响应

2.6.3　函数文件

函数文件是用来定义子程序的。它与程序文件的主要区别有三点：

(1) 由 function 起头，后跟的函数名必须与文件名同；

(2) 有输入输出变元(变量)，可进行变量传递；

(3) 除非用 global 声明，程序中的变量均为局部变量，不保存在工作空间中。

先看一个简单的函数文件，其文件名为 mean. m，键入 type mean 后，屏幕将显示元件的内容

```
function y = mean(x)
% MEAN 求平均值。对于向量，mean(x)返回该向量 x 中各元素的平均值
% 对于矩阵，mean(x)是一个包含各列元素平均值的行向量
[m, n] = size(x);
    if m==1 M=n; end        % 处理单行向量
y=sum(x)/m
```

文件的第一条语句定义了函数名、输入变元以及输出变元.没有这条语句，该文件就成为程序文件，而不再是函数文件了。输入变元和输出变元都可以有若干个，但必须在第一条语句中明确地列出。

程序中的前几条带%的字符行为文件提供注解，键入 help mean 命令后，系统将显示这几条文字，作为对文件 mean. m 的说明，这和主程序文件相同。

变量 m，n 和 y 都是函数 mean 的局部变量，当 mean. m 文件执行完毕，这些变量值会自动消失，不保存在工作空间中。如果在该文件执行前，工作空间中已经有同名的变量，系统会把两者看作各自无关的变量，不会混淆. 这样，调用子程序时就不必考虑其中的变量与程序变量冲突的问题了。如果我们希望把两者看成同一变量，则必须在主程序和子程序中都加入 global 语句，对此共同变量作出声明。

给输入变元 x 赋值时，应把 x 代换成主程序中的已知变量，假如它是一个已知向量或矩阵 Z，可写成 mean(Z)，该变量 Z 通过变元替换传递给 mean 函数后，在子程序内，它就变成了局部变量 x 。

下面的例子是多输入变量函数 logspace，用于生成等比分割的数组：

```
function y = logspace(d1, d2, n)
% logspace 对数均分数组
% logspace(d1, d2)在 10^d1 与 10^d2 之间生成长度为 50 的对数均分数组
% 如果 d2 为 pi，则这些点在 10^d1 和 pi 之间
% logspace(d1, d2, n)生成的数组长度为 n，n 的缺省值为 50
    if nargin == 2 n = 50; end              % 输入变元分析及 n 的缺省值设置
    if d2 == pi d2 = log10(pi); end         % d2 为 pi 时的设置
    y=(10).^[d1+(0: n-2)*(d2-d1)/(n-1), d2]; % 将结果返回到输出变元
```

在本例中使用了特定变量 nargin 表示输入变元的数目。当只有两个输入变元时，它默认地设定 n=50，nargout 表示输出变元数目的变量。MATLAB 常常根据 nargin 和 nargout 的数目不同而调用不同的程序段，从而体现它的智能作用。

再来看 MATLAB 是如何定义一个任意非线性函数的。在对微分方程作数值积分或解任意非线性方程求解时，都需要先列出一个这样的函数文件。一个典型程序如下：

```
function y = humps(x)
% humps  由 QUADDEMO、ZERODEMO 和 FPLOTDEMO 等程序调用的一个函数
% humps(X) 是一个在 x = 0.3 and x =0 .9 附近有尖锐极大值的函数
% 参看 QUADDEMO、ZERODEMO 和 FPLOTDEMO
   y = 1 ./ ((x-.3).^2 + .01) + 1 ./ ((x-.9).^2 + .04) - 6;
```

程序中的运算都采用元素群算法，以保证此函数可按元素群调用。MATLAB 中几乎所有的函数都能用元素群运算，所以我们自编的子程序，也要尽量满足元素群算法的要求。

2.6.4 文件编辑器及程序调试

MATLAB 提供的编辑器的用法与 Word 相仿，同时，它把编辑与调试结合在一起了。实际上，MATLAB 的主程序是比较好调试的，因为 MATLAB 的查错能力很强，配上工作空间中变量的保存和显示功能，不需要用专门的调试命令，调试也可以很方便地进行。

需要用调试命令的主要是函数程序。因为在函数程序中出错而停机时，其变量不作保存。虽然它也会指出出错的语句，但因为子程序中的变量在程序执行完毕后会自动消失，其他现场数据都无记录，这会给调试带来很大困难。解决这个问题可用下列措施：

(1) 把某些分号改为逗号，使中间结果能显示在屏幕上，作为查错的依据。

(2) 在子程序中适当部位加 keyboard 命令，到了此处，系统会暂停而等待用户键入命令。这时子程序中的变量还存在于工作空间中，可以对它们进行检查。

(3) 将函数文件的第一行前加%号，使它成为程序文件，作初步调试。第一行中的输入变元，可改用 input 或赋值语句来输入，调好后再改回函数文件。

(4) 使用 MATLAB 提供的调试命令。其命令共有 11 条，可参见第 3 章中的表 3-1。根据我们的体会，当程序不太长(例如 20 行以下)时，用调试命令反而麻烦。所以在入门课程中不予讨论。关于调试命令的用法，可参阅参考文献[1]。

第 3 章 MATLAB 的开发环境和工具

作为一种优秀的计算软件，MATLAB 之所以能够成功，不仅因为其语法和编程的简洁高效，而且在于它有一个便于使用开发并与其他软件(甚至硬件)进行交互通信的环境。比如它可以在多种计算机机型和操作系统下运行，其使用的界面和程序又完全相同，使程序设计与平台无关；它能够和一些重要的文字编辑器及图形编辑器进行交互，把计算结果很方便地组织成为图文并茂的文章等等。从 MATLAB3.x 起，其语言已经相当成熟，版本的每一次升级，在语言方面的变化很少，变动主要集中在工具箱的扩充和开发环境的改善。这方面的内容涉及面很广，不可能在一本教科书中叙述清楚。本章将以 MATLAB 6.x 版本为主要对象，对此做一扼要介绍。

3.1 MATLAB 与其他软件的接口关系

3.1.1 与磁盘操作系统的接口关系

1. 变量的存储和下载

save 命令把工作空间中的全部变量值存入磁盘，其默认的文件名是 matlab.mat。第二次再用 save 命令时，如果仍用默认文件名，则原来文件中的数据就被冲销，所以通常都要自设文件名。如果只要把 a、b、c 三个变量保存在名为 aa.mat 的文件中，则可键入

save aa a b c

mat 格式用户是读不懂的。如果要保存为 ASCII 码格式，则应再加上一个格式说明符

save aa a b c —ascii

load 是 save 的逆过程，它把磁盘上存储的 mat 数据文件取回到 MATLAB 工作空间中。其默认的文件名也是 matlab.mat。在不用默认文件或默认格式时，其命令格式与 save 命令相仿，惟一的差别是它不能选择变量。例如 load aa，它把 aa.mat 文件中的全部数据连同其变量名都下载到工作空间中。

格式说明符还有多种，MATLAB 6.x 及 5.x 的默认格式与 MATLAB 4.x 不同。因此，在 MATLAB 4.x 下存入的 mat 格式变量不能为 MATLAB 6.x 直接读出，必须在读命令的后面加上特殊的格式说明－v4，例如 load aa－v4。读者在遇到此问题时可从 help save 或 help load 中寻找详细说明。

表 3－1 列出了 MATLAB 的通用命令。

表 3－1　通用命令库(general)(f)

类别	命令	功能	命令	功能
函数的管理命令	what	列出 M、MAT 和 MEX 文件	which	找函数和文件所在的子目录
	type	显示 M 文件的全部内容	pcode	建立微码文件(P 文件)
	edit	编辑 M 文件	inmem	列出内存中的函数
	lookfor	在求助文字中搜索关键词	mex	编译 MEX 函数
通用信息	help	在线帮助文件	whatsnew	未列入说明书的新功能信息
	helpwin	有独立视窗的在线帮助文件	readme	显示 readme 文件
	helpdesk	超文本帮助文件	ver	MATLAB 和各工具箱的版本
	demo	运行演示程序		
工作空间管理	who	列出工作空间变量	save	从工作空间存储变量到磁盘
	whos	列出工作空间变量详情	clear	从内存中清除变量和函数
	load	从磁盘取出变量到工作空间	pack	紧缩工作空间内存
管理搜索路径	path	查找和改变 MATLAB 搜索路径	rmpath	在搜索路径上去除子目录
	addpath	在搜索路径上增加子目录	editpath	修改搜索路径
文件操作系统	cd	更改当前工作目录区	pwd	显示当前工作目录
	dir	列出子目录	web	打开 Web 浏览器
	delete	删除文件	computer	当前的计算机型号
	getenv	获取环境参数	Ctrl C	中断 MATLAB 的运行
命令窗控制	profile	设置 M 文件执行时间	format	设置输出格式
	clc	清除命令窗	diary	保存 MATLAB 运行文字记录
	home	使光标复原到左上角	more	在命令窗中控制分页输出
启动退出	quit	退出 MATLAB	matlabrc	启动的主 M 文件
	startup	启动 MATLAB 时的 M 文件		
公共信息	info	关于 Mathworks 公司的信息	hostid	MATLAB 服务主顾的识别码
	subscribe	订购 MATLAB 须知		
调试命令	dbstop	设置断点	dbclear	清除断点
	dbcont	继续执行	dbdown	改变局部工作空间内容
	dbstack	列出谁调用谁的清单	dbstatus	列出所有断点的清单
	dbstep	执行一行或几行	dbtype	列出带行号的 M 文件
	dbup	改变局部工作空间内容	dbquit	退出调试模式
	dbmex	调试 MEX 文件	dbstop	停止调试

2. 工作日志的记录

diary 命令可把 MATLAB 工作过程中的全部屏幕文字和数据以文本方式记录下来，成为一个工作记录，默认的文件名为 diary。因为它是文本文件，并可由任何文字处理器来修改编辑，所以有很大的使用价值，其用法如下。

当准备做记录时，在命令窗中键入 diary on 或 diary bbb，后者用 bbb. txt 为文件名。从此时开始，所有在 MATLAB 命令窗中出现的文字和数据都将记录在 diary. txt 或 bbb. txt 文件中。当需记录的过程结束，应键入 diary off。此后的屏幕内容即不做记录。如果再次使用 diary on 或 diary 文件名，则新记录的内容将接在原记录的后面，不会冲销原记录。diary 文件可以用 Notepad 或 WinWord 打开阅读。

为了避免在日志文件中记录不必要的调试过程和“垃圾内容”，应该在程序调试成功、运行无误后再打开日志文件，让程序正式运行一次。有时还需先键入 echo on，使得被执行的语句也在屏幕上显示并被记录到日志中去。记录中如发现有不必要的内容，可用文字处理器予以删改。diary 文件不能记录 MATLAB 运行中生成的图形。

3. 日期和时间命令

MATLAB 中的某些命令是与操作系统有内在联系的。除了前面说过的它可直接应用的操作系统命令 dir、delete、cd 等之外，有关时间和日期方面的命令，都是从操作系统中提取数据的。这些命令见表 3－2。

表 **3－2** 时间和日期函数库(**timefun**)(**w**)

当前日期	now	当前日期和时间的时间数	clock	当前日期的日期向量
	date	当前日期的字符串		
基本函数	datenum	成序列的日期数	datevec	日期向量
	datestr	日期的字符串格式		
日期函数	calender	日历	comday	月末日的星期数
	weekday	星期数	datetick	日期的格式设定
定时函数	cputime	以秒计的 CPU 时间	etime	经历时间
	tic，toc	秒表定时器的启动和停止	pause	暂停等待时间

下面介绍如何确定做某种计算所需的时间。例如，想看看做 1 个 100×100 阶矩阵的求逆运算所需的时间，可以用下列三组语句之一：

(1) t0 ＝clock；y＝inv(rand(100，100))；etime(clock，t0)

(2) t＝cputime；y＝inv(rand(100，100))；cputime－t

(3) tic；y＝inv(rand(100，100))；toc

这三种方法的差别在于：第一种方法要先后两次提取年月日时分秒数据并将他们相减；第二种方法以开机时间为基准；第三种方法则用 tic 把秒表置零，求得的 toc 就是经历的时间。

4. 不退出 MATLAB 环境运行其他软件

以“!”开始的命令表示这是一个 DOS 操作系统的命令。可以用这个方法在不退出 MATLAB 环境的条件下，运行以 DOS 操作系统为基础的其他软件。

3.1.2 与文字处理系统 Winword 的关系

1. 利用剪贴板进行交互

MATLAB 的程序要利用文字处理系统来编辑修改，它的运行结果(包括数据和图形)需要由图文处理系统来整理加工，因此它与 Word 图文处理系统有非常紧密的关系。它的命令窗中的所有文字数据及图形窗中的所有图形都可用 Windows 的剪贴板(Clipboard)送到 Word 中去，并可以用 Word 对它们进行编辑，形成图文并茂的书面报告。

在图形窗中截取图形时，应先用鼠标拖动边缘的方法将图形窗调到需要的大小，然后用鼠标单击菜单中的【Edit】项，在【Copy Options】子项中有【Metafile】(矢量模式)和【Bitmap】(点阵模式)。通常应选【Metafile】，因为这种模式便于在 Word 中做进一步的缩放修改。在设定完毕后，再选定【Copy Figure】或【Copy】，图就放到剪贴板上去了。然后，可把这个图贴向 Word 的任何文本文件并在其中做进一步的编辑修改。在 MATLAB 中缩放可以保持图中标注文字的可读性，而在 Word 中缩放图形往往会使文字排列不好。所以，建议在 MATLAB 中先把图形比例取得大体合适，避免到 Word 中做大幅度的缩放调整。

2. 文字编辑器的使用

在 MATLAB 6.x 中，已经把 Word 中的文字编辑功能集成为 MATLAB 的程序编辑和调试器。在图 1 - 2 显示的命令窗中，按下最左边的按钮，就会激活其程序编辑和调试器，生成图中的视窗。该视窗中的各个按钮的形式和功能与 Word 界面的几乎完全相同，所以不必细说。它的特殊之处在于：

(1) 它会用不同颜色显示 MATLAB 规定的保留字(蓝)、非法字符(鲜红)、注释字符(绿)、引用字符(深红)等。

(2) 存储文件名的后缀为.m，即生成的是 M 文件。

(3) 当被编辑的文件以 function 开头，即被编辑的是一个函数文件时，MATLAB 编辑器会自动将存储文件名定为该程序中的函数名(见第 2.6 节中函数文件的命名规定)。

(4) 能对程序自动缩进排版，便于阅读和调试。选定需要排版的程序段，单击菜单项【Text】下的子项【Smart Indent】，即可完成。

(5) 它有程序调试器功能，反映在菜单项【Debug】的各子项中。

3. Notebook 软件工具

Notebook 是 Mathworks 公司开发的软件，它在 Word 和 MATLAB 两个软件系统之间搭起了一座双向接口的桥梁。当这个软件工作时，可在 Word 中输入含有部分 MATLAB 语句的文本文件。以后只要选中这些语句，再键入 Ctrl - Enter，该软件就会把这些语句送给 MATLAB 去执行，然后把运行的结果又送回 Word，并用不同的颜色显示输出和输入的不同。利用这个工具，教师可以边写教案，边检验教案中的程序语句。科技工作者也可一边写论文，一边让论文中的程序运行结果直接出现在论文中，不再需要来回剪贴了。不过要运行这个工具，必须在安装 MATLAB 时，把 Notebook 软件工具装入系统。

3.1.3　图形文件的转储

可以把 MATLAB 的图形文件转储为多种标准图形格式，以便用各种图形软件进行处理。存储时所用后缀可以是各种标准图形格式的后缀，如 gif、bmp、jpg 等。它们可由图形窗对图形进行存储而得到。在 MATLAB 5.1 及以前的版本中，图形的存储用 print 命令来完成。例如，先激活一个图形窗，然后在命令窗中键入

```
print -dbitmap figaa
```

这就会在当前的目录下存入一个文件名为 figaa.bmp 的图形文件。要知道其他格式的命令，可以参阅 help print。

在 MATLAB 6.x 版本中，除了用 print 命令外，还可用菜单操作来实现图形转储。只要单击图形窗的菜单项【File】的子菜单【Export】(导出)，就会出现图 3-1 的界面。在【Save as Type】中选定存储格式，给出文件名，再单击【Save】，即可完成图形的存储。这里用【Export】表示 MATLAB 把图形转储为其他软件的格式，是软件之间的接口转换。这样生成的文件不属于 MATLAB 文件的范畴。

图形窗上还有一排图标组成的工具栏，可以完成图形的编辑、缩放、旋转、加字符标注等功能，将鼠标移至该图标上时，会出现其功能的说明，此处无须一一介绍，读者可以在试用中学习。

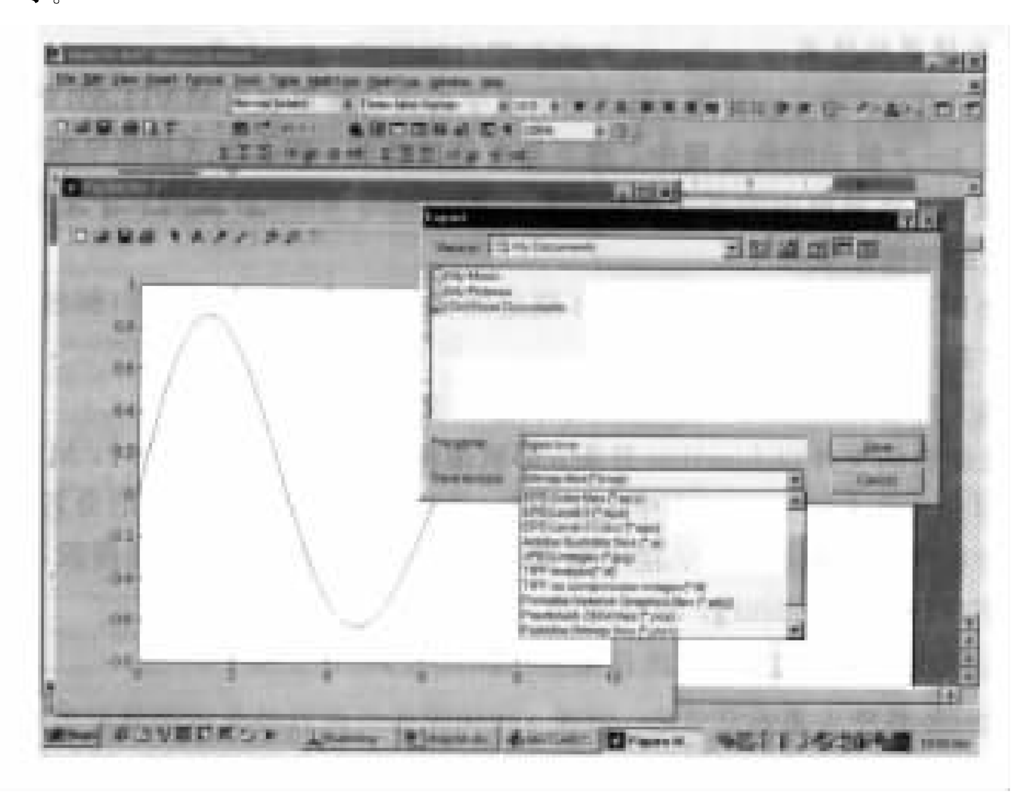


图 3-1　MATLAB 6.x 的图形窗及其转储(导出)界面

3.1.4　低层输入/输出函数库

MATLAB 可以用 load 和 save 命令来保存和提取数据，其数据可以是 mat 或 ASCII 码格式，这已在前面讲过。但这只适合于 MATLAB 环境自身。作为一种科学计算软件，与其他软件系统进行直接的(没有人参与的)数据交换是十分重要的，它可以避免人为差错和

运行低效。通过输入输出文件进行数据交换是有效的方法之一。因为几乎任何算法语言都有有限的几种输入输出文件格式(例如二进制格式和 ASCII 码字符格式),MATLAB 可以用这几种格式进行读写,也就保证了它可以在这一级上与其他语言相连接。例如,将其他软件产生的或仪器测量的数据自动读入 MATLAB,再进行分析处理并绘成图形输出等。读不同格式的文件要用不同的命令,这个库中的命令见表 3-3。

表 3-3　低层文件输入/输出库命令(iofun)(j)

文件开闭及 I/O	fopen	打开文件	fscanf	从文件读入格式化数据
	fclose	关闭文件	fprintf	把格式化数据写入文件
	fread	从文件读入二进制数据	fgetl	从文件读入一行,去掉新行字符
	fwrite	把二进制数据写入文件	fgets	从文件读入一行,保留新行字符
文件定位	ferror	询问文件 I/O 的出错状态	ftell	提取文件位置指针
	feof	测试文件结尾	frewind	倒回文件
	fseek	设置文件位置指针		
字符串及文件名处理	sprintf	把格式化数据写入字符串	sscanf	从字符串中读取格式化数据
	MATLABroot	MATLAB 安装的根目录	partialpath	部分路径名
	filesep	本平台的目录分割符	mexext	本平台的 MEX 文件名后缀
	pathsep	本平台的路径分割符	fullfile	从各部分构成全文件名
	tempdir	获取当前目录	tempname	获取当前文件
文件输入输出	load	从 MAT 文件下载到工作空间	save	把工作空间变量存入 MAT 文件
	dlmread	从 ASCII 码分隔文件读取矩阵	dlmwrite	把矩阵写入 ASCII 码分隔数据文件
	wk1read	读 WK1 文件	wk1write	在 WK1 格式的文件中写入矩阵
图像声音 I/O	imread	从图形文件读出图像	imfinfo	返回图像文件的信息
	imwrite	把图像存入图形文件		
	wavwrite	写入 WAVE(".wav")声音文件	wavread	读出 WAVE(".wav")声音文件

如果要在一个二进制文件 aaa.bin 中写入工作空间中的变量 x,则其程序为如下两条语句:

fid1=fopen('aaa.bin','r+');　% 打开 aaa.bin,'r+'表示可读可写,fid1 为文件标识

N=fwrite(fid1,x,'float')　% 将 x 以 float(浮点)格式写入 fid1 文件,返回实际写入的元素数 N

从数据文件读出变量是一个逆过程。例如，要从 aaa.bin 读入二进制数据并将它赋值给 A，程序可编写如下：

```
frewind(fid1)
fid1=fopen('aaa.bin','r+');
A=fread(fid1,[5,5],'float')
```

注意到这个程序比写入时多了第一行，因为文件的读写犹如磁带，写入以后必须倒带才能重放，要先键入倒带命令 frewind(fid1)，而第三句表示将 fid1 文件中的前 25 个数据以 float(浮点)格式读出，列成 5×5 阶矩阵，赋予变量 A。如果以后还有从 fid1 文件读出的语句，就将从第 26 个数据开始。

输入输出的格式必须相同。MATLAB 内部本来只有一种双精度格式，现在要变换为其他语言中的多种数据类型，所以会很不适应。读者应在学了 C 语言或其他语言后再来理解本节。库中每个命令的具体用法可参看 help 文本，此处不多占篇幅。

在进行音频信号或图像处理时，需要与声音文件及图像文件接口。MATLAB 也提供了相应的命令，可参看表 3－3。

在 MATLAB 中还有动态数据交换的函数库(dde)。利用它可以不经过"文件"这个中间环节而直接把运行 MATLAB 的计算机和运行其他软件的计算机通过网络进行数据交换，使 MATLAB 与其他软件平台之间的双向调用成为可能。这个函数库中的内容见表 3－4。

表 **3**－**4** 客户机函数库(**dde**)(**g**)

动态数据交换	ddeadv	建立链接	ddereq	从应用取得数据
	ddeexec	送出执行字符串	ddeterm	结束 DDE 对话
	ddeinit	DDE 对话初始化	ddeunadv	卸除链接
	ddepoke	把数据送到应用		

3.1.5 与 C 和 FORTRAN 子程序的动态链接

MATLAB 本身是用 C 语言编写的，它的丰富的科学计算子程序库中的许多经典部分来自久经考验的 FORTRAN 程序库。它可以直接调用经过一定的处理后的 C 和 FORTRAN 可执行文件，因而使执行这些子程序的速度与 C 语言及 FORTRAN 语言相同。这些可执行文件就是后缀为 mex 的文件。除了 MATLAB 中已有的 mex 文件外，用户也可把自己找到的其他可执行文件加入系统。

MATLAB 高级工具箱中还有 C 编译器，可把 MATLAB 语言编写的子程序编译成 C 语言程序，以期提高它的运行速度。MATLAB 6.x 是用 Java 语言扩展的，因此也为它今后充分利用 Java 的功能创造了有利条件。

3.2 MATLAB 的文件管理系统

3.2.1 安装后的 MATLAB 文件管理系统

如果用光盘来安装 MATLAB 软件，不管版本有何差别，其过程和其他软件相仿，此处从简。安装后的 MATLAB 根目录(通常表为 MATLABroot)下，至少有 bin、extern、help、toolbox 这四个子目录，其中子目录 bin 包含了 MATLAB 所要用到的二进制文件。启动 MATLAB 的执行文件 matlab.exe 就在这个目录中，双击这个文件就可以启动 MATLAB 软件。子目录 extern 包含了 MATLAB 所要用到的外部文件。子目录 help 包含了 MATLAB 的各种帮助文件，如果有下一级子目录 pdf_doc，则其中将包括 MATLAB 及其工具箱的说明书，那是十分有用的资料。子目录 toolbox 包含了 MATLAB 的各种函数库及已装入的作为下一级子目录的工具箱名称等，它至少应有 local 和 matlab 两项，其中 matlab(注意用的是小写)又有 22 个子目录，分别是本书第 1～4 章介绍的 MATLAB 中的基本函数集，如 datafun、elfun 等。通常在 MATLAB 根目录下，还会自动建立一个用户的子目录 work，以便把用户自编的程序存在这个子目录下，免得与系统中原有的文件系统混淆。

3.2.2 MATLAB 自身的用户文件格式

MATLAB 的用户文件通常包括以下几类：

· 程序文件　包括主程序和函数文件，其后缀为.m，即 M 文件。通常它由文本编辑器生成。MATLAB 的各个工具箱中的函数，大部分也是 M 文件。

· 数据文件　其后缀为.mat。在 MATLAB 命令窗中，用 save 命令存储的变量，在默认条件下就生成这类文件。

· MATLAB 的可执行文件　其后缀为.mex。它们由 MATLAB 的编译器对 M 文件进行编译后生成。其运行速度远高于直接执行 M 文件的速度。

此外，用 Simulink 工具箱建模，会生成模型文件(后缀为.mdl)和仿真文件(后缀为.s)，这些是 MATLAB 自身的文件格式。

3.2.3 文件管理和搜索路径

MATLAB 管理的文件范围由它的搜索路径来确定。该搜索路径由 MATLAB 启动文件来规定。其中有一段程序列出了所有由它管理的文件目录名称(在 MATLAB 6.x 中，这段程序写成名为 pathdef.m 的子程序)，这名称要列到最低层子目录。例如，MATLABroot toolbox matlab elfun。当然，这些子目录不只限于 MATLAB 根目录下的范围，整个计算机资源管理器文件系统中的任何一个底层文件夹，都可以列入 MATLAB 的搜索路径，在这些文件夹中的文件都可以被执行。反之，如果用户编写的程序未存入 MATLAB 搜索路径的子目录中，则 MATLAB 将找不到它，因而也无法运行这个程序。

要显示或修改搜索路径，可以用 path 命令。

- path：列出 MATLAB 的搜索路径。
- path(path，'新增路径名')：在原搜索路径群中加入一个新路径。

例如，在 C 盘中已有一个子目录 user1，要把它放入 MATLAB 的搜索路径上，可键入 path(path，'c:　user1')。注意这个子目录必须先建立好，使用 path 命令才有效，否则 MATLAB 找不到这个子目录，它会显示“无此子目录，命令无效”，并拒绝执行。

用命令行设置路径要键入文件夹的全路径名，很容易出错，用菜单工具较好。MATLAB 6.x 所设置的修改路径的菜单工具如下：

在命令窗中点击【File】→【Set Path】，就会出现图 3－2 所示的【Set Path】对话框。该对话框左侧是一排按钮，包括【Add Folder...】【Add with Subfolders...】、【Move to Top】、【Move Up】、【Remove】、【Move Down】和【Move to Bottom】等。如果要将某文件夹(连它的子文件夹)都列入 MATLAB 搜索路径上去，可点击【Add with Subfolders】，此时将弹出一个系统文件搜索框，即图 3－2 上右下角的小框。在其中找到该文件夹，选中它，再按【确定】，小框即关闭。然后再在【Set Path】对话框中下面一横排按钮中，按【Save】和【Close】按钮即可。

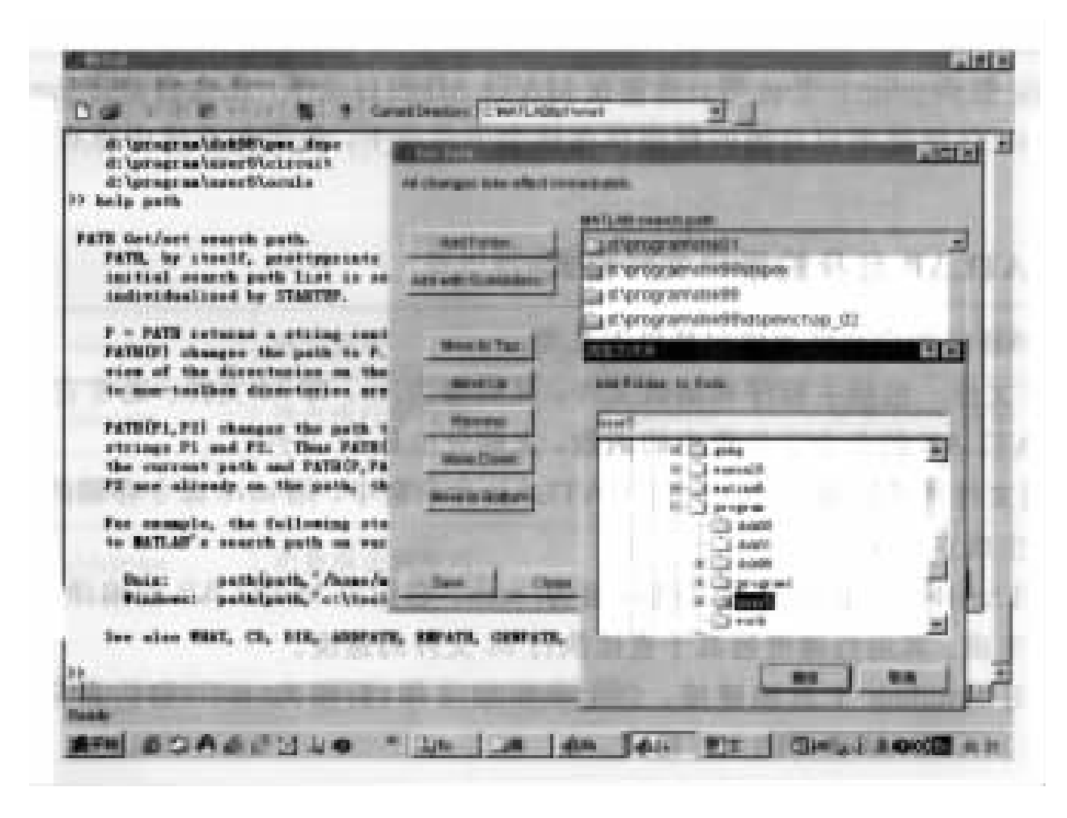


图 3－2　MATLAB 6.x 中修改搜索路径的对话框

3.2.4　与目录和搜索有关的命令

- dir：列出当前目录下的文件和子目录名。
- cd：改变当前目录，如要往上改，用 cd ..；如要往下改，用 cd [下一级子目录名]。
- delete：删除某个文件。

说明：三个都是 DOS 操作系统的命令，在 MATLAB 中同样有效。

- what [子目录名]：列出该子目录下的 MATLAB 自身的文件名，包括：

后缀为 m 的 MATLAB 程序文本文件；

后缀为 mex 的 MATLAB 二进制执行文件；

后缀为 mat 的 MATLAB 的数据文件；

后缀为 mdl 的 MATLAB 的仿真模型文件；

…

· which [文件名]：显示该文件所在的子目录路径，便于查看或修改它。例如，键入

which path

则显示

c:　matlab　toolbox　matlab　general　path.m

说明：path 命令在通用函数的库(general)中。

· lookfor [字符串]：在全部 help 文件中搜索包含该字符串的内容。例如，想找到所有与等高线绘制有关的命令，可键入

lookfor contour

得

```
CLABEL Add contour labels to a contour plot.
CONTOUR Contour plot.
CONTOUR3 3-D contour plot.
CONTOURC Contour computation.
MESHC Combination MESH/CONTOUR plot.
SURFC Combination SURF/CONTOUR plot.
```

3.2.5　搜索顺序

在 MATLAB 执行程序时，当遇到一个字符串时，如何判别该字符串的意义呢？它按如下的顺序(优先级)与已有的记录相比较：工作空间的变量名→内部固有变量名→.mex 文件名→.m 文件名。如果两个名字相同，它只认优先级高的名字。例如，用户在工作空间中给 i 赋了值，那么系统就不会取内部固有变量中设定的虚数 i；如果用户在程序中设立了一个与 MATLAB 函数同名的变量，则每次调用此名字时，出现的将是用户自定的变量，调不出 MATLAB 中的函数。所以用户在自设变量名时要防止与 MATLAB 中的函数重名。

MATLAB 中也有函数同名只是后缀不同的情况。因为 mex 后缀是二进制的执行文件，它的运行速度比.m 文件快得多，所以会优先执行它。.mex 文件通常是对.m 文件编译后生成的，因此无法阅读也不好修改。

3.3　MATLAB 6.x 的开发环境

3.3.1　桌面系统的内容

第 1 章中初步介绍了 MATLAB 的几个基本视窗。随着系统的升级，它们在不断升级，而且为了开发者的方便，不断增加新视窗。到 MATLAB 6.x 则发展到一个新阶段，它把

多种开发工具集成为MATLAB桌面系统。该系统由桌面平台以及组件组成，包含如下八个组成部分：命令窗口(Command Window)、历史命令窗口(Command History)、资源目录本(Launch Pad)、当前路径浏览器(Current Directory Brower)、帮助浏览器(Help Browser)、工作空间浏览器(Workspace Browser)、数组编辑器(Array Editor)以及程序编辑调试器(Editor-Debugger)。它们的功能简述如下：

(1) 命令窗口：第2章中的全部工作都是在命令窗中完成的，所以不必更多解释。

(2) 历史命令窗口：用于记录并显示本次工作进程中曾键入的全部行命令。利用它可以方便地修改和输入较长的行命令，或把多个有用的行命令挑选出来，组成一个完整的程序文件。因此，这是一个很有用的工具。

(3) 资源目录本：用于把用户在当前系统中安装的所有MATLAB产品说明、演示以及帮助信息的目录集成起来，便于用户迅速调用查阅。在MATLAB 7.0中，取消了这个窗口。

(4) 当前路径浏览器：用于随时显示系统当前目录下的MATLAB文件信息，包括文件名、文件类型、最后修改时间以及该文件的说明信息等。

(5) 帮助浏览器：所有的帮助信息都可以在该浏览器中显示。而且用户可以对原有的帮助信息编辑取舍，或加入自己的注解，形成自己的帮助文件。

(6) 工作空间浏览器：用于显示所有目前保存在内存中的MATLAB变量的名称、数学结构、字节数以及类型，并与按下工作空间查看按钮或键入whos命令所得的结果相同。只是在工作空间浏览器中，还可以对变量进行编辑或图形操作。

(7) 数组编辑器：用户可以直接在数组编辑器中修改所打开的数据，甚至可以更改该数据的数学结构以及显示方式。

(8) 程序编辑调试器。以上各组件都独立地构成视窗，具有自己的菜单和工具条，可以对视窗中的内容进行编辑和存储，这就使它们的功能更强大，使用更方便。对初学者而言，太多的视窗只会造成混乱，因此在本书第1章中，我们只介绍最基本的几个视窗，现在才做较详细的讨论。即便如此，如果自己不在应用中去实践，学了也很难记住，所以本书只能做简略介绍，读者仍需自己看说明书并实际应用，才能真正掌握。

3.3.2 桌面命令菜单简介

图1-3的第一行给出了MATLAB 6.x的桌面命令菜单区，它包括【File】、【Edit】、【View】、【Web】、【Window】、【Help】等六项。在第六项的右边，增加了一个显示当前目录的信息区。在主菜单上增加了Web项，表明它在联网功能上的加强。它的其他功能扩展主要反映在子菜单中。

在【File】下的子菜单中，增加了【Import Data...】(数据导入)、【Save Workspace As...】(将工作空间保存为文件)、【Set Path...】(搜索路径设定)、【Preference...】(选择)等选项。

在【Edit】下的子菜单各项，与一般文本编辑命令相仿，此处从简。只有【Paste Special】选项有些特别。用它可打开数据输入向导，将剪贴板的数据输入到MATLAB工作空间中。

在【View】下的子菜单中，增加了【Desktop Layout】(桌面布局)、【Undock Command Window】(与命令窗分离)、【Command Window】、【Command History】、【Current Directory】、【Workspace】、【Launch Pad】、【Help】、【Current Directory Filter】以及【Workspace

View Options】等选项，用以选定观察的视窗。

在子菜单项【Desktop Layout】之下又有下一级子菜单，利用它可以同时显示两个以上的视窗。显示方案列在下一级子菜单中，分别为【Default】(默认方式，同时显示【Commend Window】、【Launch Pad】和【Commend History】三个视窗)、【Command Window Only】、【Simple】(同时显示【Commend Window】和【Commend Histroy】或【Workspace】两个视窗)、【Short History】、【Tall History】(各显示【Commend Window】、【Current Directory】和【Commend History】三个视窗，但形状和排列不同)以及【Five Panel】(同时显示五个视窗)等。

【Web】是 MATLAB 6.x 新增的菜单项，通过该菜单项可以直接得到 MATLAB 的网络资源，当然此时系统必须在联网状态下。

【Window】菜单项提供了在已打开的各 MATLAB 视窗之间的切换功能，也可以用它关闭全部视窗。

【Help】下的子菜单项【Full Product Family Help】、【MATLAB Help】、【Using the Desktop】、【Using the Command Windows】和【Demo】分类提供了进入各类帮助信息系统的入口。

3.3.3　MATLAB 6.x 的用户界面

在 Windows 的资源管理器中，双击 MATLAB　bin　matlab.exe 文件，或者双击已经处于 Windows 桌面上的 MATLAB 图标，就可以启动 MATLAB 软件环境。它首先显示出一个标志 MATLAB 6.x 的图形标志，经过几秒钟后，屏幕上将出现图 3－3 所示的 MATLAB 桌面平台，它由命令窗口(Command Window)、历史命令窗口及资源目录本等三个视窗组成。根据用户的需要，可以选定并激活相应的视窗进行操作。例如，在上述界面中关闭后两个视窗，或在【Desktop Layout】中选定【Command Window Only】，就会出现图 1－3 所示的单一命令窗。

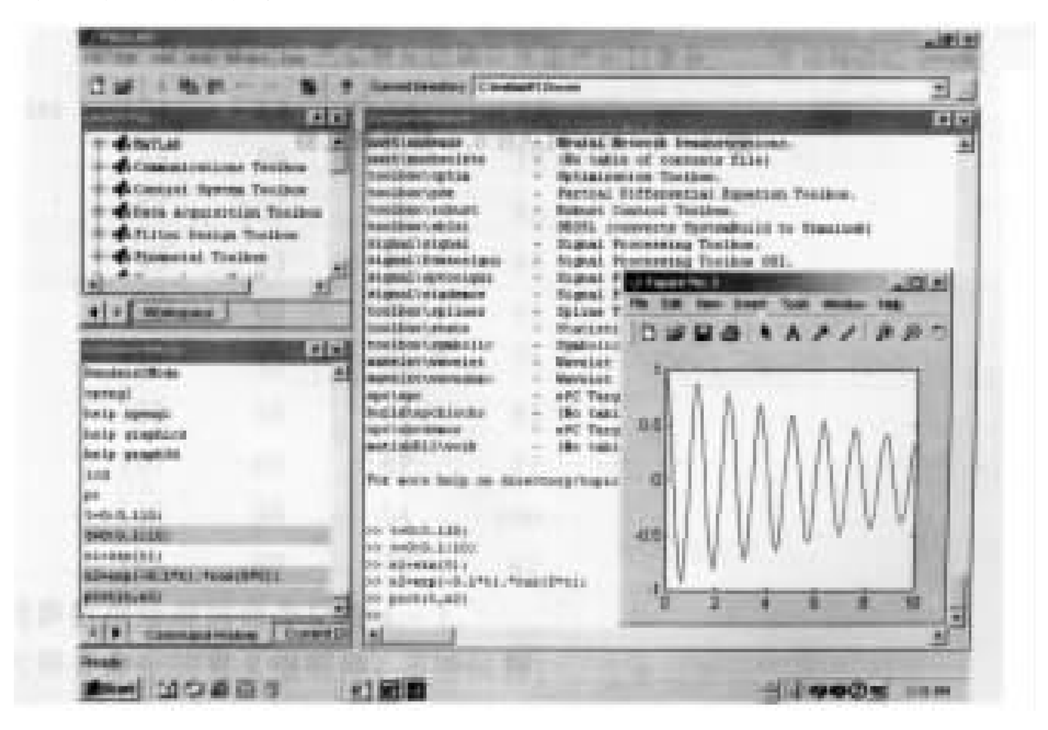


图 3－3　MATLAB 6.x 在默认条件下的桌面平台及历史命令窗的应用

图 3－4 中的图形窗是由命令窗中的 plot 语句打开的。画出这个衰减正弦曲线的有效命令有三条，要想把这三条命令组织起来，形成一个程序文件，就可以利用历史命令窗。如图 3－4 所示，在历史命令窗中用 Ctrl＋单击鼠标左键，依次选出这三条语句，进行【Copy】，并将他们【Paste】到文本编辑器中去，再【Save】起来即可。也可以把它们【Paste】到命令窗中直接执行。

再来看看 MATLAB 6.x 的帮助浏览器。在【Vicw】下选择 Hclp 项，就会出现图 3－4 所示的帮助浏览器，左边是目录栏，右边是帮助的内容。在这个图上可以看到，目录中有三个 angle 命令，它们出现在不同的工具箱中，图 3－4 中显示的是第一个 angle 命令的意义和用法。另外，浏览器的目录部分还给出了多种搜索与查找方法。



图 3－4　MATLAB 6.x 帮助浏览器

第 4 章　MATLAB 的其他函数库

在前三章中，我们已介绍了 MATLAB 基本部分的 21 个函数库中的 12 个，这些内容实际上已远超过了 FORTRAN 和 C 语言在科学计算上的性能，对于大学低年级的初学者来说，已经足以应付各种大学课程算题的需要。余下的这几个函数库中，有些是涉及深一些的数学和物理概念，对低年级同学而言有些超前，不易很快接受，可以留到高年级再深入；有些则是用来编写较高级的程序用的，要对 MATLAB 编程比较熟悉后再学。单独把这些内容放在一章中，可以避免对初学者造成拦路虎。这一章内容具有独立性，其各节也有独立性，读者可以跳过整章或整节，向后阅读。但要读懂本章各节，必须以前三章为基础，并具备相应的数学和理论知识。

4.1　数据分析和傅里叶变换函数库（datafun）

4.1.1　基本的数据分析

MATLAB 的基本数据处理功能是按列向进行的，因此要求待处理的数据矩阵按列向分类，而行向则表示数据的不同样本。例如，10 个学生的身高及三门课程分数列表如下：

```
data = 154    49    83    67
       158    99    81    75
       155   100    68    86
       145    63    75    96
       145    63    75    96
       141    55    65    75
       155    56    64    85
       147    89    87    77
       147    96    54   100
       145    60    76    67
```

进行简单数据处理的命令见表 4－1。

其中大部分命令的意义很明确，不需解释。

std 标准差是指列中 N 个元素与该列平均值 mean(data)之差的平方和开方。即

$$\mathrm{std(data)} = \sqrt{\sum_{N}(\mathrm{data}-\mathrm{mean(data)})^2}$$

表 **4－1** 一些数据处理命令的结果

命　令	功　能	身　高	课程 1	课程 2	课程 3
max(data)	求各列最大值	158	100	87	100
min(data)	求各列最小值	141	49	54	67
mean(data)	求各列平均值	149.2	73.0	72.8	82.4
std(data)	求各列标准差	5.7504	20.4070	10.0421	12.0757
median(data)	求各列中间元素	147	63	75	81
sum(data)	求各列元素和	1492	730	728	824
trapz(data)	梯形法求积分	1342.5	675.5	648.5	757.0

trapz 求积分可以看成求和，梯形法求积分近似于求元素和，其差别在于梯形法是把相邻两点数据的平均值作为数据点。10 个数据只能产生九个数据点，相加以后比元素和少一组数据，把它乘以步长才真正表示了面积。其差额为半个首点和半个末点的数据和，即

$$\mathrm{trapz}(data) = \mathrm{sum}(data) - 0.5(data(1) + data(N))$$

在 N 很大时两者的误差很小。

有些数据处理命令的结果不是一个标量而是一个列向量，为了节省篇幅，我们只取数据中的前三行，其结果见表 4－2。注意其结果一般是与原数据具有同样的行数，只有求差分(diff)会减少一行，因为它是求相邻行之间的差。另外，cumtrapz 函数是用梯形法累计求面积，和 trapz 相仿，它也会使数据长度减少一个。

表 **4－2** 产生列结果的数据处理命令

命　令	功　能	身　高	课程 1	课程 2	课程 3
cumsum(data(1:3，:))	列向累加和	154	49	83	67
		312	148	164	142
		467	248	232	228
cumprod(data(1:3，:))	列向累乘积	154	49	83	67
		24332	4851	6723	5025
		3771460	485100	457164	432150
diff(data(1:4，:))	列向差分	4	50	－2	8
		－3	1	－13	11
		－10	－37	7	10
Sort(data(1:3，:))	列向重新排序	154	49	68	67
		155	99	81	75
		158	100	83	86
cumtrapz(data(1:4,:)) *	列向累加积分(相当于不定积分)	156.0000	74.0000	82.0000	71.0000
		312.5000	173.5000	156.5000	151.5000
		462.5000	255.0000	228.0000	242.5000

4.1.2　用于场论的数据分析函数

用于场论的命令有以下几个：

· gradient：用来求二维和三维场的近似梯度，例如根据电位分布求电场就可用这个函数。

· del2：是二维和三维场的拉普拉斯算子。

· cross：为两个向量的矢量积。

· dot：为两个向量的数量积。

设 $\boldsymbol{i}$，$\boldsymbol{j}$，$\boldsymbol{k}$ 为沿 x，y，z 方向的单位向量，则对于两向量 $\boldsymbol{a}=a_x\boldsymbol{i}+a_y\boldsymbol{j}+a_z\boldsymbol{k}$ 和 $\boldsymbol{b}=\boldsymbol{b}_x\boldsymbol{i}+b_y\boldsymbol{j}+b_z\boldsymbol{k}$ 而言：

向量的矢量积为(叉乘)

$$a\times b=(a_yb_z-a_zb_y)\boldsymbol{i}+(a_zb_x-a_xb_z)\boldsymbol{j}+(a_xb_y-a_yb_x)\boldsymbol{k}$$

向量的数量积为(点乘)

$$a\cdot b=a_xb_x+a_yb_y+a_zb_z$$

在 MATLAB 中这两个向量可表为：

```
a=[ax，ay，az]；b=[bx，by，bz]；
cross(a，b) = [ay * bz-az * by，az * bx-ax * bz，ax * by-ay * bx]；
dot(a，b) = a * b'；
```

4.1.3　用于随机数据分析的函数

MATLAB 有两个产生随机数的命令：rand(m，n)产生在 0 与 1 之间均匀分布的 m 行 n 列随机数矩阵，其均值为 0.5，标准差(或均方根差)为 0.2887；randn(m，n)产生正态分布的 m 行 n 列随机数矩阵，其均值为 0，标准差为 1；其分布情况可用直方图命令 hist(x，N)来显示。其中 N 表示直方图横坐标的分割数，其默认值为 10。例如

```
x=rand(1，1000)；hist(x)
y=randn(1，1000)；
```

得出的两组图形如图 4－1 所示。hist(x)是把 1000 个 x 中处于 0～0.1，0.1～0.2，…，0.9～1.0 各个区域中的数目分别清点出来，画成直方图。如果 x 真是均匀分布的，那么这

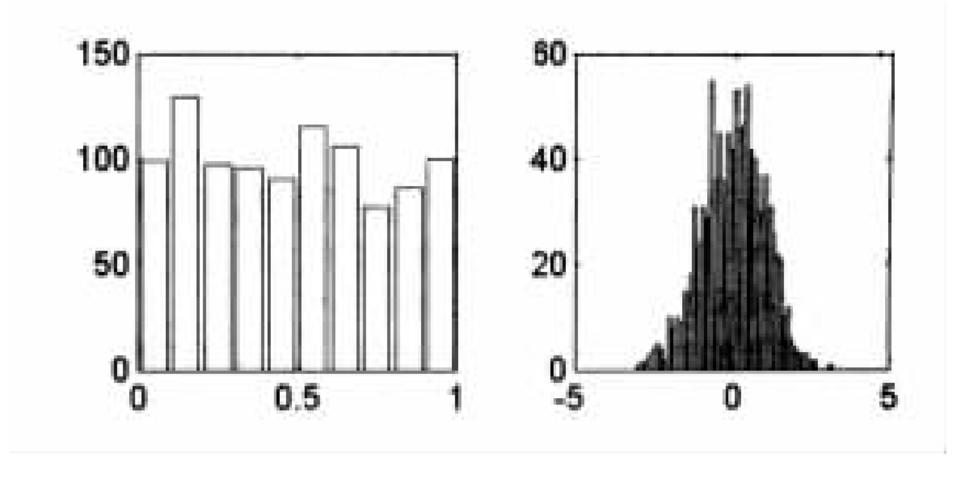


图 4－1　均匀分布与正态分布随机数直方图

个图应该是水平直线。实际上，随机数规律是按统计方法确定的，所以各区域的数量仍参差不齐，只有数据量无限增加时，此规律才愈益接近。hist(y，50)则把 y 的最小值和最大值之间分成 50 份进行统计，得到一个钟形的，即正态分布的曲线。

4.1.4 用于相关分析和傅里叶分析的函数

相关分析(包括卷积)和傅里叶分析分别用于信号的时域和频域处理。这里虽然只给出了十几个函数，实际上它们是整个信号处理计算的基础。

(1) corrcoef 给出两个同长信号的相关系数，例如对前面两个随机序列，键入

R＝corrcoef(x，y)

得　R ＝ [1　　　0.0508
　　　　−0.0508　　1]

对角线上是 x 和 y 的自相关系数，这说明 x 和 y 自相关性很强，而互相关则很弱。

(2) cov(x，y)给出 x，y 的协方差矩阵，对上述 x，y，有

cov(x，y) ＝ [0.0785　　−0.0148
　　　　　　−0.0148　　1.0782]

其主对角线上的值分别为 x 和 y 的均方差，即标准差的平方(因为是随机数，它不会严格等于理论值)。

(3) conv(x，y) 给出 x，y 的卷积。如果 x 是输入信号，y 是线性系统的脉冲过渡函数，则 x，y 的卷积就给出系统的输出信号。卷积函数也用于多项式相乘，见下节。

(4) filter(b，a，x)也是根据输入信号 x 和线性系统特性求输出信号的函数。其不同在于系统的特性是以传递函数的分子多项式系数向量 b 和分母多项式系数向量 a 给出，而不是以脉冲过渡函数的形式给出的。

(5) X＝fft(x，N)求出时域信号 x 的离散傅里叶变换 X。N 为规定的点数。N 的默认值为所给 x 的长度。当 N 取 2 的整数幂时变换的速度最快。通常取大于又最靠近 x 的幂次，即令 N＝2 ^ nextpow2(length(x))。例如 x 的长度为 12，nextpow2(12)＝4，N＝2^4＝16，多出的各点补以零。一般情况下，fft 求出的函数为复数，可用 abs 及 angle 分别求其幅度和相位。在画频谱图时往往最关心其幅频特性。

【例 4－1－1】 给出一个信号

t＝0：.001：3；u＝sin(300 * t)＋2 * cos(200 * t)

它的幅频特性可用下列语句求得

U＝fft(u)；plot(abs(U))

得出的频谱曲线如图 4－2(*a*)所示，它对采样频率呈对称形式。为了把它看得更清楚，把坐标缩小，键入 axis([0，300，0，3000])，得出的频谱曲线如图 4－2(*b*)所示。

(6) x＝ifft(X)为傅里叶反变换函数，其用法与 fft 相仿。

(7) sound(u，s)会在音箱中产生 u 所对应的声音。s 规定重放的速度，其缺省值为 8192 (b/s)，见表 4－3。

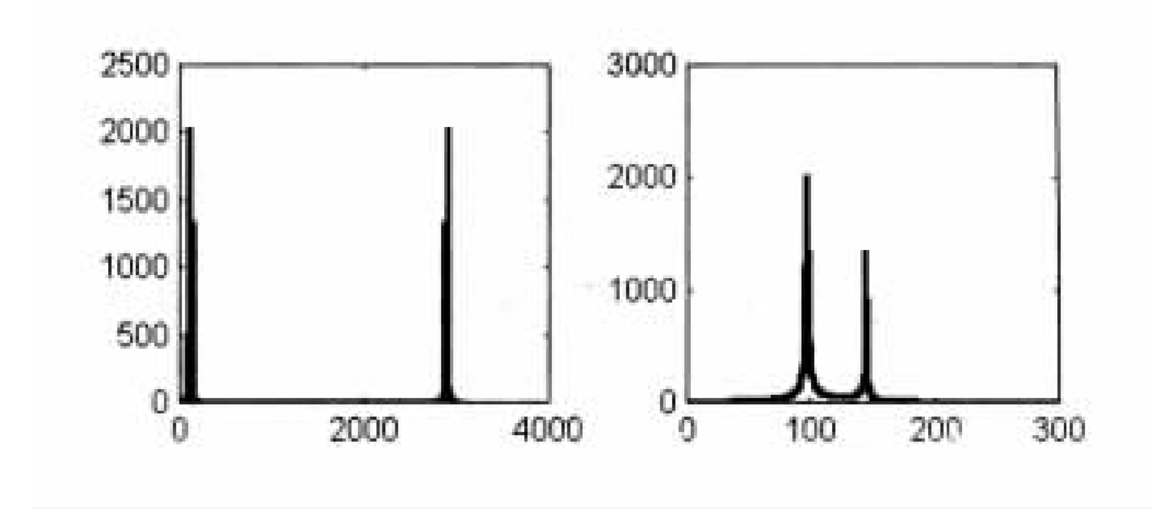


图 4-2 例题中信号的幅频曲线

表 4-3 数据分析和傅里叶变换函数 **mat5(d)**

基本运算	max	最大元素	sum	元素之和
	min	最小元素	prod	元素之积
	mean	平均值	cumsum	元素的累加和
	median	中间值	cumprod	元素的累积
	std	标准差	hist	直方图
	sort	按升序排列	trapz	用梯形法作定积分
	sortrows	按升序排列行	cumtrapz	梯形法作不定积分
差分	diff	差分函数和近似微分	gradient	近似梯度
	del2	五点离散拉普拉斯算子		
相关运算	corrcoef	相关系数		
	cov	协方差矩阵		
滤波和卷积	filter	一维数字滤波	filter2	二维数字滤波
	conv	卷积和多项式相乘	conv2	二维卷积
	convn	n 维卷积	deconv	反卷积和多项式相除
傅里叶变换	fft	离散傅里叶变换	ifft	离散傅里叶反变换
	fft2	二维离散傅里叶变换	ifft2	二维离散傅里叶反变换
	fftn	n 维离散傅里叶变换	ifftn	n 维离散傅里叶反变换
	fftshift	将零迟延移到频谱中心		
声音函数	sound	把向量放成声音	mu2lin	把 mu—规律编码变为线性信号
	soundsc	自动设比把向量放成声音	lin2mu	把线性信号变为 mu—规律编码

4.2　矩阵的分解与变换函数库（matfun）

4.2.1　线性方程组的系数矩阵

在 2.2 节中我们提到了可以用矩阵除法来解线性方程组，本节将讨论有关解线性方程组的一些深入的问题及其工具函数。这些函数见表 4－4 中的矩阵分析和线性方程部分。

表 **4－4**　矩阵函数和数值线性代数(**matfun**)(**m**)

矩阵分析	norm	矩阵或向量的范数	null	零空间正交基
	normest	矩阵 2 范数的估值	orth	正交化
	rank	矩阵的秩	rref	缩减行梯次格式
	det	行列式(必须是方阵)	subspace	两个子空间之间的夹角
	trace	主对角线上元素的和		
线性方程	和 /	线性方程求解	qr	正交三角分解
	chol	Cholesky 分解	cholinc	不完全 Cholesky 分解
	cond	矩阵条件数	condest	1 范数条件数的估值
	rcond	linpack 逆条件数计算	nnls	非负最小二乘
	lu	高斯消去法系数矩阵	pinv	矩阵伪逆
	inv	矩阵求逆(必须是方阵)	lscov	协方差已知的最小二乘
特征值和奇异值	eig	特征值和特征向量	eigs	若干特征值
	poly	特征多项式(必须是方阵)	condeig	对应于特征值的条件数
	polyeig	多项式特征值问题	schur	Schur 分解
	hess	Hessenberg 形式	balance	均衡(改善条件数)
	qz	广义特征值	svd	奇异值分解
矩阵函数	expm	矩阵指数	expm2	用泰勒级数求矩阵指数
	expm1	用 M 文件求矩阵指数	expm3	用特征值求矩阵指数
	logm	矩阵对数	funm	通用矩阵函数的计算
	sqrtm	矩阵开方		
分解工具	qrdelete	从 QR 分解中删去一列	rsf2csf	实对角阵变为复对角阵
	qrinsert	在 QR 分解中插入一列	cdf2rdf	复对角阵变为实对角阵
	planerot	Given's 平面旋转		

det(a)用以求方阵 a 的行列式。若 det(a)不等于零，则 a 的逆阵 inv(a)存在。线性方程组的系数矩阵只有满足这个条件，它的解才存在。rank(a)用以求任意矩阵 a 的秩，也就是它所能划分出的行列式不为零的最大方阵的边长。trace(a)求出矩阵主对角线上元素的和。

如果 det(a)虽不等于零，但数值很小，近似于零，则这样的线性方程组称为病态的。其解的精度比较低。为了评价线性方程组系数矩阵的病态程度，用了条件数(Condition Number)的概念。条件数愈大，方程病态愈重，解的精度愈低。若系数矩阵的条件数很大而又拿它作除数(即解此线性方程组)时，MATLAB 会提出警告："条件数太大，结果可能不准确"，求条件数的函数为 cond(a)，a 可以不是方阵。

在线性方程组 A * x=B 中，系数矩阵 A 的行数 n 表示方程的数目，其列数 m 表示未知数的数目。在正常情况下，方程数等于未知数数，即 n=m，A 为方阵，A　B 意味着 inv(A) * B。实际上，对于方程数大于未知数数(n>m)的超定方程组，以及方程数小于未知数数(n<m)的不定方程组，MATLAB 中 A　B 的算式都仍然合法。前者是最小二乘解；而后者则是令 x 中 m−n 个元素为零的一个特殊解。在这两种情况下，因为 A 不是方阵，其逆 inv(A)不存在，解的 MATLAB 算式均为 x=inv(A′ * A) * (A′ * B)。把 pinv(A)=inv(A′ * A) * A′ 定义为伪逆函数，则 A　B 就等于 pinv(A) * B。因此 MATLAB 中引入了伪逆的概念，除数 A 就可以不是方阵。

【例 4-2-1】　求下列矩阵的行列式及逆阵等特性。

```
a = 2    9    0    0
    0    4    1    4
    7    5    5    1
    7    8    7    4
```

得

```
det(a) = -275
rank(a) =  4
inv(a) =  -0.0727    0.4255    0.7855   -0.6218
           0.1273   -0.0945   -0.1745    0.1382
           0.0000   -0.6000   -0.8000    0.8000
          -0.1273    0.4945    0.3745   -0.3382
trace(a) = 15
cond(a) = 33.4763
```

4.2.2　矩阵的分解

矩阵可以分解为几个具有特殊构造性质的矩阵的乘积，这是分析矩阵的一种重要手段，而这种分解在数学计算上通常又是非常繁琐的工作。MATLAB 提供了一些现成的函数可供调用，主要有以下三种。

(1) 三角分解(lu 分解)。它把一个任意方阵分解为一个准下三角方阵和一个上三角方阵的乘积。由于此函数有两个输出矩阵，其左端应有两个变量 l 和 u，键入

```
[l, u]=lu(a)
```

得

```
l = 0.2857    1.0000         0         0
         0    0.5283    0.6838    1.0000
    1.0000         0         0         0
    1.0000    0.3962    1.0000         0
u = 7.0000    5.0000    5.0000    1.0000
```

```
0    7.5714   -1.4286   -0.2857
0         0    2.5660    3.1132
0         0         0    2.0221
```

l 被称为准下三角阵，是因为必须交换两行才能成为真的下三角阵，它的行列式绝对值等于 1(可正可负)，上三角阵 u 的行列式等于 a 的行列式。lu 分解只能对方阵进行，它常用于高斯消去法。

(2) 正交分解(qr 分解)。qr(a)把任意矩阵 a 分解为一个正交方阵 q 和一个与 a 有同样阶数的上三角矩阵 r 的乘积。该方阵 q 的边长为矩阵 a 的 n 和 m 中之小者，且其行列式的值为 1。

【例 4－2－2】 求下列 3×5 矩阵 b 的 qr 分解。

```
b = 0.2190    0.6793    0.5194    0.0535    0.0077
    0.0470    0.9347    0.8310    0.5297    0.3834
    0.6789    0.3835    0.0346    0.6711    0.0668
```

键入

```
[q, r]=qr(b)
```

得

```
q = -0.3063   -0.4667   -0.8297
    -0.0658   -0.8591    0.5076
    -0.9497    0.2101    0.2324
r = -0.7149   -0.6338   -0.2466   -0.6886   -0.0911
          0   -1.0395   -0.9490   -0.3390   -0.3189
          0         0   -0.0011    0.3805    0.2038
```

(3) 奇异值分解(svd 分解)。svd(a)把任意 n×m 矩阵 a 分解为三个矩阵的乘积，即 a=u＊s＊v。其中 u，v 分别为 n×n 和 m×m 的正交方阵，s 则为 n×m 的对角阵。对角线上的元素就是矩阵 a 的奇异值，其长度为 n 和 m 中的小者。例如对上述 b 作奇异值分解，键入

```
[u, s, v]=svd(b)
```

得

```
u =  0.4623    0.2273    0.8571
     0.7822    0.3507   -0.5149
     0.4176   -0.9085    0.0157
s = 1.7539         0         0         0         0
         0    0.7995         0         0         0
         0         0    0.3534         0         0
v = 0.2403   -0.6885    0.4927   -0.4748         0
    0.6872    0.1674    0.3027    0.4193    0.4819
    0.5158    0.4729    0.0506   -0.3723   -0.6076
    0.4102   -0.5151   -0.6122    0.3194   -0.2994
    0.1890    0.0944   -0.5369   -0.5985    0.5558
```

矩阵最大奇异值和最小奇异值之比就是它的条件数。即

```
cond(b)=max(diag(s))/min(diag(s))
```

4.2.3 矩阵的特征值分析

eig(a)用来求方阵 a 的特征根和特征向量，其输出有两个，即特征向量 e 和特征根 r。键入

```
[e, r]=eig(a)
```

得

```
e = -0.2568    -0.3834 + 0.4681i   -0.3834 - 0.4681i    0.6167
    -0.3481    -0.2177 - 0.2869i   -0.2177 + 0.2869i   -0.1850
    -0.4682     0.5152 + 0.2228i    0.5152 - 0.2228i   -0.6624
    -0.7705     0.4217 - 0.1060i    0.4217 + 0.1060i    0.3829
r =  14.2004    0                   0                   0
     0          0.7495 + 5.2088i    0                   0
     0          0                   0.7495 - 5.2088i    0
     0          0                   0                  -0.6993
```

特征根是特征方程的根，矩阵的特征方程系数可用 poly 函数求出，再用 roots 命令也可求出其特征根。例如，键入

```
p=poly(a)
```

得

```
p =   1.0000   -15.0000   38.0000   -359.0000   -275.0000
```

而

```
roots(p) =   14.2004
              0.7495 + 5.2088i
              0.7495 - 5.2088i
             -0.6993
```

结果与 eig 函数求出的特征根相同，但 roots 函数不能求特征向量。

4.2.4 特殊矩阵库（specmat）

有一些特殊构造的矩阵在矩阵变换中很有用处，MATLAB 4 将它们组成一个专门的函数库。读者在应用中遇到有关的矩阵，即可在此调用。其调用的参数和调用方法均可从 help 文本中查得，此处不多占篇幅。MATLAB 5 已把这个库并入 elmat 库中，见表 2－1 中的“特殊矩阵”栏。

4.3 多项式函数库（polyfun）

一元高次代数多项式 $a(x) = a_1 x^n + a_2 x^{n-1} + \cdots + a_n x + a_{n+1}$，在 MATLAB 中可以用它的系数向量

a=[a(1), a(2), …, a(n), a(n+1)]

来表示。其方次已隐含在系数元素离向量右端的元素的间隔中。要注意，如果 x 的某次幂的系数为零，这个零必须列入系数向量中。在微分方程中，通过运算微积可把线性系统的特性表示为两个算子 s 的多项式之比。因此多项式在近代信息和控制理论中有着十分重要的地位。MATLAB 的多项式和插值函数见表 4－5。

表 4－5 多项式和插值函数(polyfun)(r)

多项式	roots	多项式求根	polyfit	用多项式曲线拟合数据
	poly	按根组成多项式	polyder	多项式及求导数
	polyval	多项式求值	conv	多项式相乘，卷积
	polyvalm	矩阵作变元的多项式求值	deconv	多项式相除，反卷积
	residue	部分分式展开(留数)		
数据插值	interp1	一维插值(一维查表)	interpn	n 维插值(二维查表)
	interp1q	快速一维线性插值	interpft	用 FFT 方法的一维插值
	interp2	二维插值(二维查表)	griddata	网格数据生成
	Interp3	三维插值(二维查表)		
样条函数插值	spline	三次样条函数插值	unmkpp	提供分段多项式的细节
	ppval	分段多项式计算	table1	一维插值表
	mkpp	构成分段多项式	table2	二维插值表
几何分析	delaunay	Delaunay 三角化	voronoi	Voronoi 图
	dsearch	最近的 Delaunay 三角化	inpolygon	点在多边形内时为真
	tsearch	最近的三角搜索	rectint	矩形相交区域
	convhull	突壳	polyarea	多边形区域
工具	xychk	向一维和二维数据变元检查	abcdchk	检查矩阵 A，B，C，D 的一致性
	xyzchk	向三维数据变元检查	ss2tf	状态空间变为传递函数
	xyzvchk	向三维容量数据变元检查	ss2zp	状态空间变为零极增益
	automesh	如果输入是自成网格的为真	tf2ss	传递函数变为状态空间
	mkpp	制造分段多项式	tf2zp	传递函数变为零极增益
	unmkpp	提供分段多项式的细节	tfchk	传递函数正确性检查
	resi2	求重极点的留数	zp2ss	零极增益变为状态空间
	tzero	传送零点	zp2tf	零极增益变为传递函数
过时的函数	icubic	一维立方插值	interp6	二维最近邻居插值
	nterp4	二维双线性数据插值	table1	一维表格查找
	interp5	二维双立方数据插值	table2	二维表格查找

4.3.1 多项式的四则运算

【例 4－3－1】 设有两个多项式 $a(x)=2x^3+4x^2+6x+8$ 及 $b(x)=3x^2+6x+9$，要求对此两个多项式作如下运算：

(1) 多项式相乘(conv)：conv 函数本来是卷积(Convolution)的意思。但它也符合多项式相乘的运算规则。可以想象把系数向量 a 正常排列，而把 b 反转，先将 a(1)与 b(1)对齐：

a(1) a(2) a(3) a(4)

b(3) b(2) b(1)

把上下对应的项相乘，a(1) * b(1)得出多项式乘积 c 的最高次项系数 c(1)；把 b 右移一位，把上下对应的项相乘并求和，a(1) * b(2)＋a(2) * b(1)为次高次项系数 c(2)；依此类推，可得到乘积 c 的全部系数。这种运算和卷积运算的规则完全相同，键入

```
a=[2，4，6，8]，b=[3，6，9]，c=conv(a，b)
```

得 a = 2 4 6 8

b = 3 6 9

c = 6 24 60 96 102 72

(2) 多项式相加：MATLAB 规定，只有长度相同的向量才能相加，因此必须把短的向量前面补以若干个零元素，才能用 MATLAB 的矩阵加法运算符。键入

```
d=a+[0, b]
```

得 d = 2 7 12 17

这种手工数两个多项式的长度再补零的方法是不可取的，必须要让计算机自动完成。为此可编一个子程序 polyadd.m，其内容为：

```
function y=polyadd(x1, x2)
n1=length(x1); n2= length(x2);
if n1>n2 x2=[zeros(1, n1-n2), x2];
elseif n1<n2 x1=[zeros(1, n2-n1), x1];
end, y = x1+x2;
```

这样，多项式相加就可写成：c = polyadd(a, b)，相减可另编一个子程序，或在 polyadd 的输入变元中加负号来实现。

(3) 多项式相除：相除是相乘的逆运算，但除法不一定除得尽，会有余子式，因此键入

```
[q, r]=deconv(c, a)
q = 3 6 9
r = 0 0 0 0 0 0
```

其中，q 是商式，r 是余子式。因为用的是相乘的数据 a 和 c，恰好除净。如令 a1=a+1，则有：

```
a1 = 3 5 7 9
[q1, r1]=deconv(c, a1)
q1 = 2.0000 4.6667 7.5556
r1 = 0 0 0 7.5556 7.1111 4.0000
```

可以用商式与除式相乘，再加上余式的方法来检验

```
c1=conv(q1, a1)+r1
```

得 c1 = 6 24 60 96 102 72

与 c 相同。

4.3.2 多项式求导、求根和求值

(1) 多项式求导数(polyder)：键入

```
e=polyder(c)
```

得 e = 30 96 180 192 102

(2) 多项式求根(roots 和 poly 函数)：键入

```
ra=roots(a); rb=roots(b); rc=roots(c); ra, rb, rc
```

得 ra = -1.6506

-0.1747 + 1.5469i

```
        -0.1747 - 1.5469i
rb = -1.0000 + 1.4142i
        -1.0000 - 1.4142i
```

rc 是 ra 和 rb 的并集，这是完全可以预计到的。

由根求多项式系数是 roots 的逆运算，其函数名也是 poly，有

```
a = poly(ra); b = poly(rb)
```

从 poly 函数的用法可以看出 MATLAB 的智能特点：当 a 是向量时，poly 把它看作根来组成多项式；当 a 是方阵时，poly 用它组成方阵的特征多项式（见 4.2 节）。

(3) 多项式求值(polyval)：将多项式 a 中的自变量 x 赋予值 xv 时，该多项式的值可用

```
F = polyval (a, xv)
```

求得，其中 xv 可以是复数，而且可以是矩阵或数组，此时 polyval 对输入变元作元素群运算，这对于求线性系统的频率特性特别方便。polyvalm 则对输入的变元阵（必须是方阵）作矩阵多项式运算。

【例 4-3-2】 设 a 为系统分母系数向量，b 为系统分子系数向量，求此系统的频率响应并画出频率特性。先令频率数组 w 取线性间隔：

```
w=linspace(0, 10);              % 在 w 等于 0～10 之间按线性间隔取 100 点(默认值)
A=polyval(a, j*w); B=polyval(b, j*w);        % 分别求分母分子多项式的值
                                              % (为复数数组)
subplot(2, 1, 1); plot(w, abs(B./A)),      % 画两者元素群相除所得的幅频特性
subplot(2, 1, 2); plot(w, angle(B./A))     % 画相频特性
```

频率特性通常在对数坐标中绘制。因此输入频率数组取对数等间隔：

```
w1=logspace(-1, 1)          % 在 w1 从 10^-1～10 之间，按对数分割为 50 点(默认值)
F=polyval(b, j*w1)./polyval(a, j*w1);    % 求出这些点上的频率响应(复数)
subplot(2, 1, 1), loglog(w1, abs(F))      % 在双对数坐标中画出幅频特性
subplot(2, 1, 2); semilogx(w1, angle(F))  % 在双对数坐标(x)中画出相频特性
```

所得曲线如图 4-3 所示。

4.3.3 多项式拟合

p=polyfit(x, y, n)用于多项式曲线拟合。其中 x，y 是已知的 N 个数据点坐标向量，当然其长度均为 N。n 是用来拟合的多项式次数，p 是求出的多项式的系数，n 次多项式应该有 n+1 个系数，故 p 的长度为 n+1。拟合的准则是最小二乘法。

【例 4-3-3】 设原始数据为 x 在十一个点上测得的 y 值：

```
x=0: 0.1: 1;
y=[-0.447, 1.978, 3.28, 6.16, 7.08, 7.34, 7.66, 9.56, 9.48, 9.30, 11.2];
```

线性拟合：a1=polyfit(x, y, 1);

求出 a1 后，可求出 xi=linspace(0, 1);（即 100 个点）上的 yi1 值并绘图：

```
yi1=polyval(a1, xi); plot(x, y, 'o', xi, yi1, 'b'), pause
```

其中，原始数据用圆圈标出，而拟合曲线为蓝色。依此类推，有：

二次拟合：a2=polyfit(x,y,2); yi2=polyval(a2,xi); plot(x,y,'o',xi,yi2,'m')

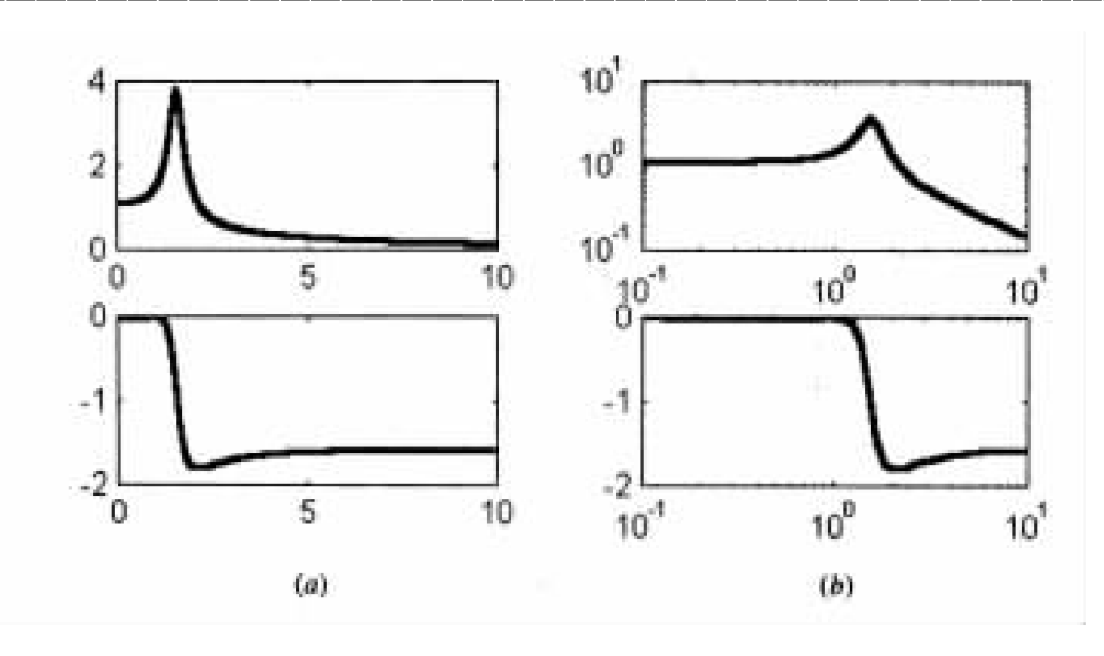


图 4－3　线性坐标和对数坐标中的频率特性

三次拟合：a3＝polyfit(x,y,3)；yi3＝polyval(a3,xi)；plot(x,y,'o',xi,yi3,'r')

九次拟合：a9＝polyfit(x,y,9)；yi9＝polyval(a9,xi)；plot(x,y,'o',xi,yi9,'c')

十次拟合：a10＝polyfit(x,y,10)；yi10＝polyval(a10,xi)；plot(x,y,'o',xi,yi10,'g')

所得的曲线如图 4－4 所示。给定十一点的最大拟合阶次为 10，此时拟合曲线将通过全部给定点。可以看出，拟合曲线的阶次太高会造成曲线振荡，反而看不出函数关系的基本规律，并不一定好。

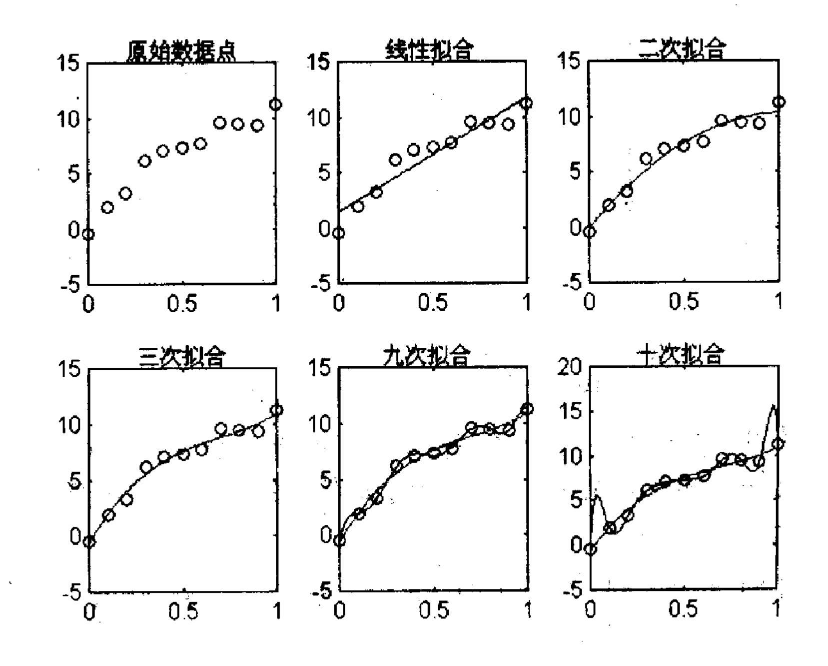


图 4－4　不同的逼近次数产生的不同曲线

4.3.4　多项式插值

插值和拟合的不同在于：(i)插值函数通常是分段的，因而人们关心的不是函数的表达式，而是插值得的数据点；(ii)插值函数应通过给定的数据点 x，y。插值函数一般地可表为

yi＝interp1(x，y，xi，'method')

其中，xi 为插值范围内的任意点集 x 坐标，yi 是插值后的对应数据点集的 y 坐标。'method'为插值函数的类型选项，有'linear'(线性，默认项)、'cubic'(三次)和'cubic spline'(三次样条)等三种。

(1) 一维插值函数 interp1。

【例 4-3-4】 仍取上例中的 x，y，xi，求其线性和三次插值曲线。

线性插值：yi1=interp1(x，y，xi)；plot(x，y，'o'，xi，yi1)

三次插值：yi2=interp1(x，y，xi，'spline')；plot(x，y，'o'，xi，yi2，'g')

所得曲线如图 4-5 及图 4-6 所示，三次插值的结果比较光滑。

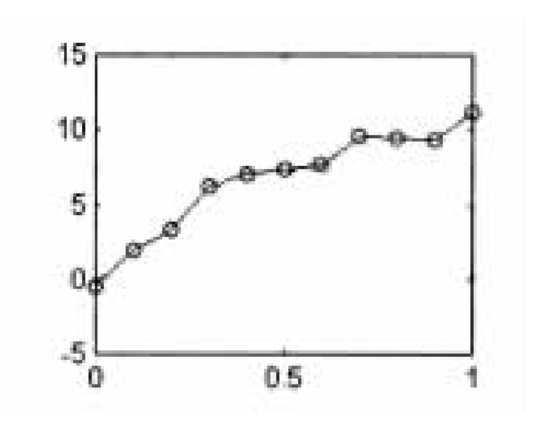


图 4-5 线性插值曲线

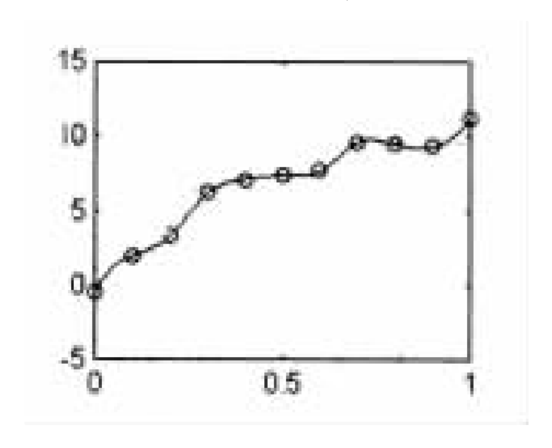


图 4-6 三次插值曲线

(2) 二维插值函数 zi=interp2(x，y，z，xi，yi，'method')，其变元的意义可以类推。

【例 4-3-5】 已知某矩形温箱中 3×5 个测试点上的温度，求全箱的温度分布。

给定：width=1：5；depth=1：3；

temps=[82 81 80 82 84；79 63 61 65 81；84 84 82 85 86]；

要求计算沿宽度和深度细分网格：di=1：0.2：3；wi=1：0.2：5；交点上的各点温度。

tc=interp2(width，depth，temps，wi，di'，'cubic')； % 求各点温度

mesh(wi，di，tc) % 画三维曲面

所得温度分布图形如图 4-7 所示。

注意 interp2 中所用的 wi 和 di'是宽度和深度方向的细分坐标向量，di 必须变换为列向量，插值函数运算时会自动将它们转变为宽度乘深度平面上的网格，并计算网格点上的温度。

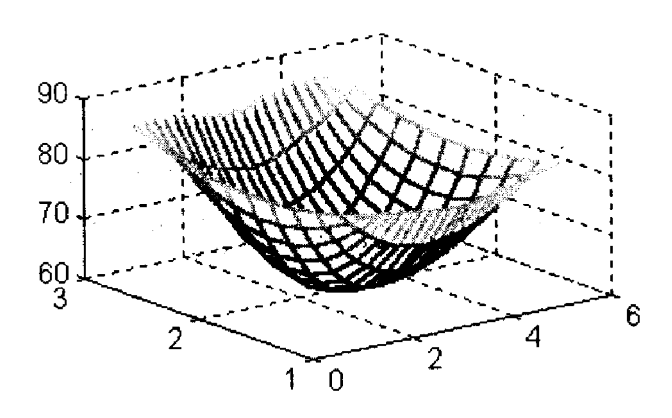


图 4－7 二维插值的曲面

4.3.5 线性微分方程的解（residue）

线性常微分方程的解可用拉普拉斯算子 s 表示为

$$Y(s)= B(s)/A(s)$$

其中 B(s)和 A(s)都是 s 的多项式，分母多项式的次数 n 通常高于分子多项式的次数 m。在时间域的解 y(t)是 Y(s)的拉普拉斯反变换。求反变换的重要方法之一是部分分式法，即将上述多项式分解为多个 s 的一次分式之和。表 4－5 中的留数函数 residue 可以完成这一任务。步骤为：

(1) 用[r，p，k]＝ residue(b，a) 求出 Y(s)的极点数组 p 和留数数组 r，因而 Y(s)可表为

$$Y(s)=\frac{r(1)}{s-p(1)}+\frac{r(2)}{s-p(2)}+\frac{r(3)}{s-p(3)}+\frac{r(4)}{s-p(4)}+\cdots$$

(2) 此时它的反变换可以很简单地求出为

y(t)＝r(1) * exp(p(1) * t)＋r(2) * exp(p(2) * t)＋r(3) * exp(p(3) * t)
＋r(4) * exp(p(4) * t)＋…

【例 4－3－6】 求解线性常微分方程

$$y'''+5y''+4y'+7y=3u''+0.5u'+4u$$

在输入 u(t)为单位脉冲及单位阶跃信号时的解析解。

解：用 Laplace 变换（脉冲输入 u(s)＝1，阶跃输入 u(s)＝1/s）

$$y(s)=\frac{3s^2+0.5s+4s}{s^3+5s^2+4s+7} * u(s)=\frac{b(s)}{a(s)}$$

在脉冲输入时的响应：

a＝[1，5，4，7]；b＝[3，0.5，4]；[r，p，k]＝residue(b，a)

得
```
r =   3.2288
     -0.1144 + 0.0730i
     -0.1144 - 0.0730i
p = -4.4548
     -0.2726 + 1.2235i
```

```
        -0.2726 - 1.2235i
k =        [ ]
```

求时域解，先设定时间数组 t=0：0.2：10；然后列出

yi=r(1) * exp(p(1) * t)+r(2) * exp(p(2) * t)+r(3) * exp(p(3) * t)；plot(t，yi)

在阶跃输入时的响应：此时分母由于乘了个 s，a 将提高一阶，右端多加一个零。

```
a=[1，5，4，7，0]；b=[3，0.5，4]；[r，p，k]=residue(b，a)
r = -0.7248
      0.0767 + 0.0764i
      0.0767 - 0.0764i
      0.5714 + 0.0000i
p = -4.4548
      -0.2726 + 1.2235i
      -0.2726 - 1.2235i
         0
k =        [ ]
ys=r(1) * exp(p(1) * t)+r(2) * exp(p(2) * t)+r(3) * exp(p(3) * t)+r(4)；
plot(t，ys)
```

所得曲线如图 4－8 所示。

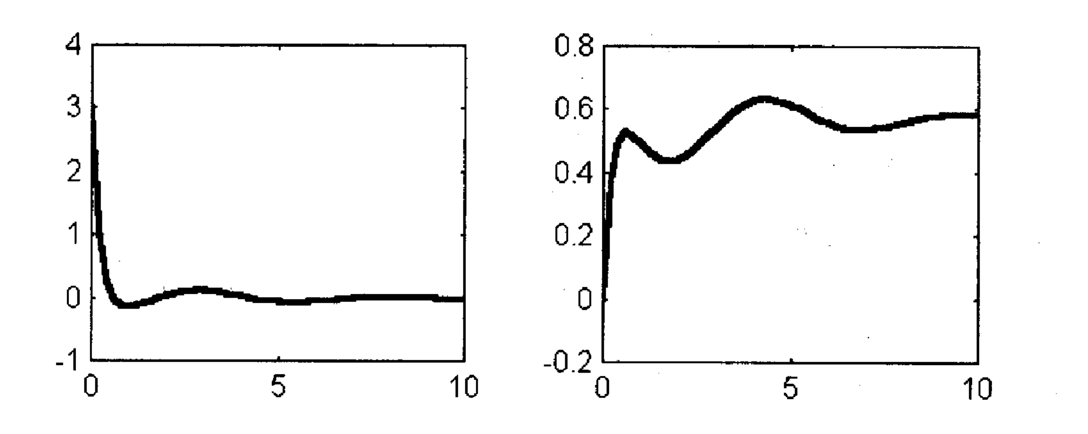


图 4－8　脉冲过渡响应和阶跃过渡响应

4.4　函数功能和数值分析函数库（funfun）

MATLAB 中的函数功能和数值分析命令有一个共同特点，其输入变元不仅有矩阵变量，而且还有函数名。在执行这些命令时，要不断地调用函数作为输入变元。这一类命令完成的功能因此比较丰富而灵活。掌握了它也就能编写出更为高明的程序了。函数功能和数值积分库见表 4－6。

表 4－6　函数功能和数值积分库(**funfun**)(**e**)

	函　数　名	功 能 说 明	需调用的函数
最优化和求根	fmin	单变量函数求极小值 ymin	给出待分析的函数 y=f(x)
	fmins	多变量函数求极小值	
	fzero	单变量函数求 y=0 处的 x	
数值定积分	quad	数值积分计算(低阶)	给出被积分的函数 f(x) dy/dx=f(x)，求 y
	quad8	数值积分计算(高阶)	
	dblquad	双精度数值积分	
函数绘图	ezplot	简便的函数绘图器	
	fplot	画函数曲线 y=f(x)	
内联(INLINE)函数对象	inline	构成 INLINE 函数对象	
	argnames	变元名	
	formula	函数公式	
	char	把 INLINE 函数转换为字符数组	
	vectorize	使字符串或 INLINE 函数向量化	
常微分方程数值积分器	ode45	解非刚性微分方程(中阶方法)	给出导数的函数表达式 dy/dx=f(y, x)，求 y
	ode23	解非刚性微分方程(低阶方法)	
	ode113	解非刚性微分方程(变阶方法)	
	ode15s	解刚性微分方程(变阶方法)	
	ode23s	解刚性微分方程(低阶方法)	
	odefile	ODE 文件语法	
ODE 输出函数	odeplot	时间序列	
	odephas2	ODE 输出函数的二维相平面	
	odephas3	ODE 输出函数的三维相空间	
	odeprint	打印 ODE 输出函数	

4.4.1　本函数库的主要子程序

本函数库的主要子程序见表 4－7，我们将它分为两类来探讨。第一类是对任意非线性函数的分析，包括求极值，过零点等；第二类是求任意函数的数值积分，包括定积分和微分方程的数值解等。它们的共同特点是必须会自行定义函数。这在本书 2.6 节中已经学过。就取其中的函数 humps.m 来说明这些子程序的用法，它定义了下列非线性函数，即

$$y=\frac{1}{(x-0.3)^2+0.01}+\frac{1}{(x-0.9)^2+0.04}-6$$

表 4－7 特殊函数(specfun)(u)

类别	函数	说明	函数	说明
特殊数学函数	airy	Airy 函数	bessely	第二类 Bessel 函数
	besselj	第一类 Bessel 函数	besselh	第三类 Bessel 函数(Hankel 函数)
	besseli	第一类修正的 Bessel 函数	besselk	第二类修正的 Bessel 函数
	beta	Beta 函数	betainc	不完全的 beta 函数
	betaln	beta 函数的对数	ellipj	Jacobi 椭圆函数
	ellipke	完全椭圆积分	erf	误差函数
	erfc	误差补函数	erfcx	标定的误差补函数
	erfinv	逆误差函数	expint	指数整数函数
	gamma	伽马函数	gammainc	不完全的伽马函数
	gammaln	伽马函数的对数	legendre	联合的 Legendre 函数
	cross	向量叉乘		
数论函数	factor	素数分解	primes	产生素数清单
	gcd	最大公约数	lcm	最小公倍数
	rat	有理分式近似	rats	有理分式输出
	isprime	是素数时为真	perms	所有可能的排列数
	nchoosek	N 取 K 的组合数		
坐标变换	cart2sph	从笛卡尔向球坐标变换	cart2pol	从笛卡尔向极坐标变换
	pol2cart	从极坐标向笛卡尔坐标变换	sph2cart	从球坐标向笛卡尔坐标变换

4.4.2 非线性函数的分析

(1) 绘制函数曲线 fplot。

其格式为：fplot('函数名'，[初值 x0，终值 xf])，例如，要画出 humps 函数在 x＝0～2 之间的曲线，键入

```
fplot('humps', [0, 2]), grid
```

得出图 4－9 所示曲线。

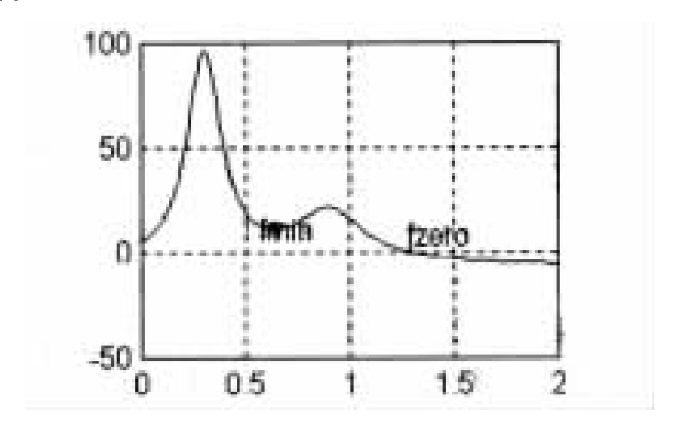


图 4－9 humps 函数的曲线

fplot 函数对于快速了解一些复杂特殊函数的波形很有用处。例如求其中第一类 bessel 函数(表 4－7)，可用

fplot('besselj(alpha, x)', [0, 10])

设 alpha 为 1，2，5 时所得曲线见图 4－10 所示。

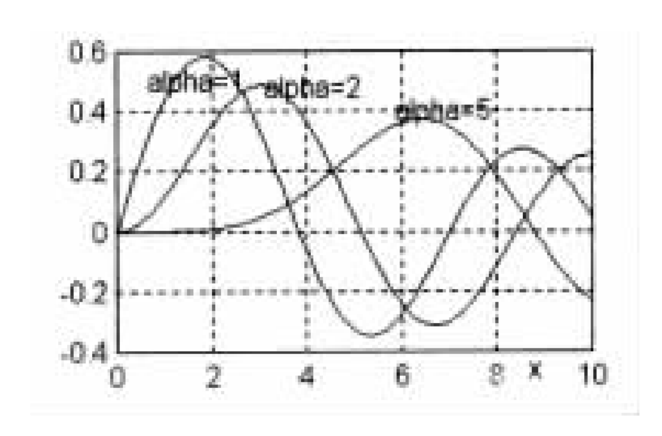


图 4－10　第一类 bessel 函数的曲线

(2) 求函数极值 fmin。

其格式为：fmin('函数名'，初值 x0，终值 xf)，例如，求 humps 函数在 x＝0～1.5 之间的极小值，键入

m＝fmin('humps', 0, 1.5)

得　　m ＝ 0.6370

(3) 求函数零点 fzero。

其格式为：fzero('函数名'，初猜值 x0)，例如，求 humps 函数在 x＝1 附近的过零点，键入

z＝fzero('humps', 1),

得　　z ＝ 1.2995

以上给出的是这些函数调用的典型格式，还有其他选项可作为变元，例如

fplot('tan', [－2 * pi 2 * pi －2 * pi 2 * pi], '*'), grid

在第二项变元中增加了 y 轴的上下限，第三项变元是线型。所得图形如图 4－11(*a*)所示，读者可从 help fplot 中得到进一步的信息。

还有一个简便画函数图的命令 ezplot(读作 easy plot)，它连自变量范围都无需规定，其默认的自变量范围为[-2π, 2π]，因此只要键入

ezplot tan(x), grid

也可得到类似于图 4－11(*a*)的曲线，只是 * 号变为了实线。若键入

ezplot tan(sin(x))－sin(tan(x))

所得图形如图 4－11(*b*)所示。可以看出，图上还自动作出了标注。

4.4.3　任意函数的数值积分

(1) 定积分子程序(quad 及 quad8)。

其格式为：quad('函数名'，初值 x0，终值 xf)，例如，求 humps 函数在 x＝1 与 2 之间

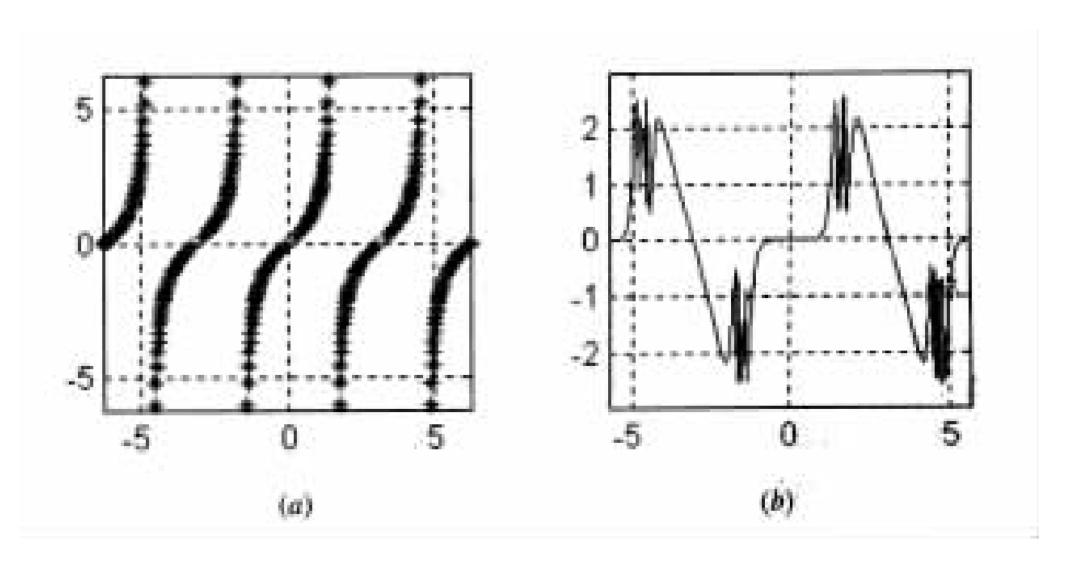


图 4－11 由 fplot 和 ezplot 画出的曲线

的定积分，键入

```
s=quad('humps', 1, 2)
```

得 s ＝ －0.5321

不难用定积分函数来求不定积分的数值解。只要固定积分下限，用 for 循环，把积分上限逐步增加即可。例如要求 humps 函数以 x=0 为下限的不定积分，可编写下列程序

```
for i=1: 20
    x(i)=0.1 * i;
    y(i)=quad('humps', 0, x(i));
end, plot( x, y)
```

得出的曲线如图 4－12 所示。可以与图 4－9 对照确认它是 humps 曲线的积分。

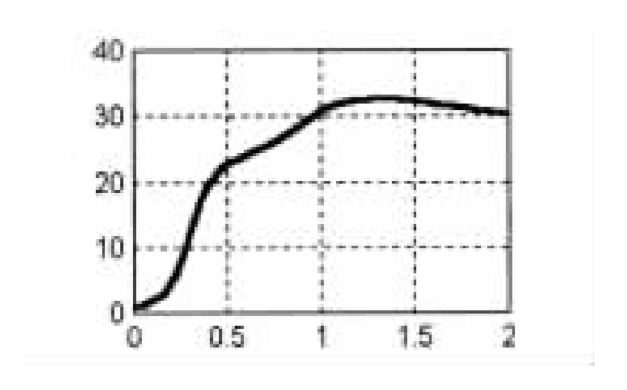


图 4－12 humps 函数的积分曲线

(2) 微分方程数字解(ode23，ode45 等)。

如果微分方程可化为一阶微分方程组的形式，即

dy/dx ＝ f(x, y)

其中，x 是标量，y 可以是一个列向量。

f(x, y)是以 x，y 为变元的函数，用 MATLAB 函数文件表述。设文件名为 yprime.m 则求此微分方程的数值解的子程序调用格式为

[x, y] ＝ ode23('yprime'，自变量初值 x0，自变量终值 xf，因变量初值 y0)

对于 humps 函数，不能直接用 ode23 作数值积分，其原因在于 humps 只有一个输入

变元 x，微分方程数值解的函数 ode23 等要求被调用的函数有两个输入变元。如果我们把 humps 函数文件加一个虚的变元 y，即把它的第一句换成 function yp = humps1(x，y)并将此函数另存成一个 humps1. m 文件，则

```
[x，y] = ode23('humps1'，0，2，1)；plot(x，y)
```

表示在初值 y0＝1 的条件下，从 x0＝1 到 xf＝2 求微分方程的数值解，则可以得到与图 4－12 相仿的曲线，只是向上平移了一个单位，因为这里设 y0＝1。在这个例子中，函数 humps1 中的 y 只是一个虚的变元，比较简单。

【例 4－4－2】 求下列微分方程(范德堡方程)的数值解，即

$$y''+r(y^2-1)y'+y=0$$

它可写成导数在左端的两个一阶微分方程构成的方程组：

$$y_1'=y_2$$

$$y_2'=r(1-y_1^2)y_2-y_1$$

先要建立反映此微分方程组右端的函数文件 vdpl. m，存入子目录 user 中，其内容为：

```
function yprime = vdpl (x，y)
global r            % r 值由主程序通过全局变量传送
yprime = [ y(2)；r * (1 - y(1).^2). * y(2) - y(1) ]；% 两行单列向量
```

主程序如下：

```
global r ，r =input('输入 r，在 0<r<10 之间选择')
x0=input('x0=')；xf= input('xf=')；y0=input('y0=[y10；y20]=')；
[x，y] = ode45('vdpl'，x0，xf，y0)；plot(x，y)
```

在 r＝2，x0＝0，xf＝30，y0＝[1；2]条件下得出的曲线如图 4－13 所示。

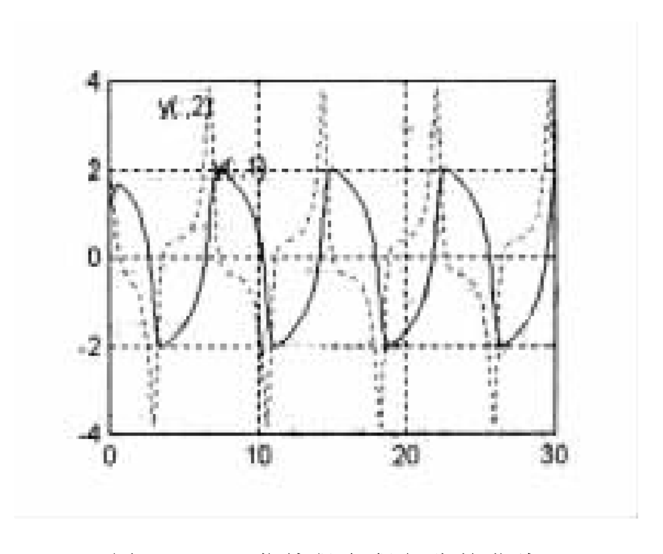


图 4－13　范德堡方程积分的曲线

ode45 是高阶的数值积分函数，其步长可以取得较大，并能保证较高的精度，调用方法与 ode23 相仿。这些函数都有自动选择步长的功能，以保证把误差控制在 0.001 以下。如果要改变容许误差 tol，可在输入变元中增加其他选项，详情可从 help 命令中获得。此外，还有 ode23s，ode15s，ode113 等数值积分函数，以及绘制相平面曲线的命令 odephas2，odephase3 等。

4.5 字符串函数库（strfun）

MATLAB 的程序和标识符都是用字符串来表示的。每个字符有它对应的 ASCII 码。4.4 节中的函数，都把字符串当作变元来看待，从而使程序更为简化和高效。有时也需要把字符串当作数码来处理。字符串函数库中的命令，都是为了增强这一功能。对初学者，这部分并不重要，但若要编写出人机界面优良，能调用各种函数和文件的高级程序，字符串函数库是必不可少的。MATLAB 的字符串函数见表 4-8。

表 4-8 字符串函数 strfun

一般函数	char	建立字符数组(字符串)	blanks	空格字符串
	double	把字符串转换为数字	deblank	去除尾部的空格
	cellstr	由字符数组组成字符阵列	eval	执行程序字符串
测试	ischar	是字符串时为真	isletter	是英文字符时为真
	iscellstr	是字符阵列时为真	isspace	是空格字符时为真
字符串比较	strcmp	字符串比较	strncmp	比较前 N 个字符串
	findstr	在字符串中找另一字符串	strjust	调整字符串
	upper	将字符串变为大写	strmatch	找到可能的匹配字符串
	lower	将字符串变为小写	strrep	用一个字符串替代另一个
	strcat	链接字符串	strtok	在字符串中找一个令牌
	strvcat	竖向链接字符串		
字符串与数的转换	num2str	把数转换为字符串	str2mat	由单个字符串形成文本矩阵
	int2str	把整数转换为字符串	sprintf	在格式控制下把数转换为字符串
	str2num	把字符串转换为数	sscanf	在格式控制下把字符串转换为数
	mat2str	把矩阵转换为字符串		
基数的转换	hex2num	把十六进制字符串转换为IEEE浮点数	dec2bin	把十进制整数变换为二进制字符串
	hex2dec	把十六进制字符串转换为十进制整数	base2dec	把基数 B 字符串变换为十进制整数
	dec2hex	把十进制整数转换为十六进制字符串	dec2base	把十进制整数变换为基数 B 字符串
	bin2dec	把二进制字符串变换为十进制整数		

4.5.1 字符串的赋值

语句 s='abxyABXY0189' 把字符串给 s 赋值，其结果是

s＝abyzABYZ0189

而　　size(s)＝ 1　12

说明它是以行向量的形式存储的。当然它内部带有字符串的标志，故在屏幕上显示出字符。要找到 s 所对应的 ASCII 码，可用 abs 命令

abs(s)＝ 97　98　121　122　65　66　89　90　48　49　56　57

从中可以知道英文大小写字母和数字的十进制 ASCII 码值。再可用 setstr 命令作逆向变换

setstr(abs(s)) ＝ abyzABYZ0189

要求出字母和数字的十六进制 ASCII 码值，可用 dec2hex 命令

dec2hex(abs(s))＝ 61 62 79 7A 41 42 59 5A 30 31 38 39

MATLAB 显示时并没有各码之间的空格，这里加上空格是为了便于读者阅读。

可以把几个字符串沿行向串接，构成更长的字符串，如键入

s1＝['welcome ', s]

得　　s1 ＝ welcome abyzABYZ0189

可以把几个长度相同的字符串沿列向并列，组成一个字符串矩阵，如

s2 ＝['a＝5 '；'b＝2 '；'c＝a＋b＊b']

这时必须在前两个字符串中增添若干个空格，保证三个字符串长度均为 7，否则赋值无效。

4.5.2　字符串语句的执行

如果字符串的内容是 MATLAB 语句，如上述的 s2，则可以用 eval 命令来使它执行。如键入

for k＝1：3 eval(s2(k，：))，end

结果为 a＝5，b＝2，c＝9。

在编写 MATLAB 的演示程序时，往往要通过人机交互，让用户输入某种表达式(而不只是数据)，然后按此表达式执行，这种程序的格式为

st ＝ input(' s＝表达式'，'s')；eval(st)

input 语句中的's'把表示输入当作字符串来接受，因此用户键入的字符串就不必加引号了。下面的例子说明了如何将 eval 函数和 load 函数一起使用，读出 10 个具有连续文件名 mydata1，mydata1，…，mydata10 的数据文件：

for i＝1：10 fname＝'mydata'；eval(['load '，fname，int2str(i)])，end

特别注意的是 int2str(i)把数 1：10 转换为字符 1：10。在显示屏上看不出两者有什么差别，但要记住字符 0：10 的 ASCII 码是 48：57。数 10 只占一个 MATLAB 双精度存储单元，而字符串 10 却占两个存储单元，并构成一个单行两列的矩阵。在 eval 后的输入变元必须是一个构成 MATLAB 语句的字符串，因此必须把三个字符串接起来，并且不要忘掉在 load 后面加一个空格。eval 命令是在较高级的程序中常常遇到的，要学会使用。

4.5.3　字符串输入输出

前面我们一直用 disp 函数来进行字符串和数据的输出。disp('pi＝')将显示引号内的

字符串，此处为 pi=。disp(pi)将显示变量 pi 的值 3.1416 或其他的八种显示格式之一，由 format 命令确定。想把字符串 pi=和变量 pi 的值显示在一行上，试用 disp('pi=', pi)，回答这是非法的。这时应该用 sprintf 函数，它可把数据按要求的格式转换为字符串，再把它与需要显示的字符串组装成一个长字符串。使显示格式非常灵活，人机界面更为友好。键入

```
st=sprintf(' 圆周率 pi= %8.5f', pi); disp(st)
```

结果为

```
圆周率 pi=3.14159
```

其中%为数据格式符，f 表示十进制浮点，8.5 表示数字的长度为 8 位，小数点后 5 位。从%到 f 之间的字符都是不显示的，它只指出显示数据 pi 的格式。

【例 4-5-1】 再举一个用 sprintf 的例子。

```
x = 0:10:90; y = [x; sin(x*pi/180)]; disp(sprintf('%10.2f %12.8f   n', y))
 0.00       0.00000000
10.00       0.17364818
  …           …
80.00       0.98480775
90.00       1.00000000
```

sprintf 命令是从 C 语言中的同名命令演化来的。sscanf 则是它的逆命令。相仿的还有 fprintf 和 fscanf(见表 3-3)。

4.6 稀疏矩阵函数库(sparfun)

在许多科学和工程计算中，人们会遇到很大的矩阵，例如几千行几千列，元素则数以百万计。而这些元素中，绝大多数是零。这反映了复杂事物中的诸变量之间，有直接关联的是少数。为了节省内存和提高计算速度，产生了稀疏矩阵的理论和方法。MATLAB 的稀疏矩阵函数库(见表 4-9)就是为这个目的开发的。稀疏矩阵只存储矩阵的非零元素，其表示形式如下例：

```
设  x =        0         0         0         0          0
               0         0         0    0.1302          0
         -0.5936         0         0         0          0
               0    1.4499    0.0388         0          0
          0.5589         0         0         0    -0.0954
```

sparse 命令把完全矩阵转换为稀疏矩阵，键入

```
s=sparse(x)
```

得

```
s = (3, 1)     -0.5936
    (5, 1)      0.5589
    (4, 2)      1.4499
    (4, 3)      0.0388
```

```
(2，4)          0.1302
(5，5)         －0.0954
```

括号内是非零元素的行号和列号，后面则是元素的值。可见 25 个元素中它只需保存六个。用命令 whos 检验 MATLAB 工作空间中的变量存储情况，结果如下：

Name	Size	Elements	Bytes	Density	Complex
s	5 by 5	6	92	0.2400	No
x	5 by 5	25	200	Full	No

表 4－9　稀疏矩阵函数(s)

初等稀疏矩阵	speye	稀疏单位矩阵	sprandn	正态分布稀疏随机矩阵
	sprandsym	稀疏对称随机矩阵	spdiags	由对角矩阵生成稀疏矩阵
	sprand	均匀分布稀疏随机矩阵		
全矩阵向稀疏矩阵的转换	sparse	从非零元素创建稀疏矩阵	full	变稀疏矩阵为全矩阵
	find	查找非零元素的下标	spconvert	稀疏矩阵的外部格式转换
用稀疏矩阵的非零元素工作	nnz	非零元素的数目	nonzeros	非零元素
	nzmax	分配给非零元素的内存总额	spones	用“1”替换非零元素
	spalloc	为非零元素分配内存	issparse	矩阵为稀疏时得真
	spfun	对非零元素施加函数	spy	显示稀疏结构
重组算法	colmmd	列的最小度	symmmd	最小对称度
	symrcm	CuthillMcKee 逆排序	colperm	按非零元素数目对列排序
	randperm	随机交换向量	dmperm	DulmageMendelsohn 分解
线性代数	luinc	不完全 lu 分解	svds	一些奇异值
	sprank	结构的秩		
线性方程(迭代方法)	pcg	预设条件的共轭梯度法	bicg	双共轭梯度法
	bicgstab	双共轭梯度稳定法	cgs	共轭梯度平方法
	gmres	通用最小残差法	qmr	准最小残差法
对树的运算	treelayout	对单树或多树的布局	Treeplot	绘出结构树
	etree	矩阵的消除树	etreeplot	绘出消除树
	gplot	按图论方法画图		
杂项	symbfact	Symbolic 符号因式分解	spparms	为稀疏矩阵设定参数
	spaugment	形成最小二乘增广系统		

可见 s 所占存储单元数为 x 所占存储单元数的 0.24 倍。

Full 命令则把稀疏矩阵转换为完全矩阵。初等矩阵运算的命令，如四则运算，求逆等均可直接用于稀疏矩阵，像矩阵分解等命令则要把它化为完全矩阵后才能调用。在大学本科中，解电路矩阵方程也会遇到大量系数为零的情况。但课程习题所遇到的阶次不会高，完全矩阵足以应付。用到稀疏函数库的可能性不大，本书只用表 4－9 把函数库列出备查。读者遇到这类应用时，得先参阅有关稀疏矩阵理论的书籍，再用 MATLAB 中的 help 文本或参看其他参考书。

4.7 图形界面函数库（uitools）

MATLAB 中的图形界面函数库是为了设计像图 1－5 和图 1－6 那样的界面而用的。通常，在设计了一个较丰富的 MATLAB 函数集之后，为了便于他人使用，应该避免让用户去记忆和键入各种各样的函数名称，做成这样的图形界面就比较方便使用。其中有计算机向用户显示结果或出错信息的各种对话框，有用户对计算机实行控制的各种按钮等。MATLAB 有一本说明书，专门介绍了图形界面的设计方法，其搜索路径为

MATLAB　help　pdf_doc　matlab　Buildgui

它所用的函数，都已包括在表 4－10 中。读者将来工作中若有需要，可以找此说明书参阅。

表 4－10　图形用户界面工具(uitools)＊(x)

GUI 函数	uicontrol	建立用户控制菜单	dragrect	用鼠标拖引方框
	uimenu	建立用户界面菜单	rbbox	橡皮擦方框
	ginput	用鼠标输入图形	waitfor	执行模块并等待事件发生
	select-movere-size	交互地选择、移动、变形或拷贝对象	waitforbut-tonpress	等待键或按钮在图上按下
	uiwait	执行模块并等待继续命令	uiresume	继续执行模块 M 文件
GUI 设计工具	guide	设计 GUI	menuedit	编辑菜单
	align	对齐 UI 控制和轴线	propedit	编辑特性
	cbedit	编辑 Callback		
对话框	dialog	建立对话图形	warndlg	警告对话框
	axlimdlg	对话框轴线范围	uigetfile	标准的打开文件对话框
	errordlg	错误对话框	uiputfile	标准的存储文件对话框
	helpdlg	帮助对话框	uisetcolor	对话框颜色选择
	inputdlg	输入对话框	uisetfont	对话框字体选择
	listdlg	列出待选对话框	pagedlg	对话框页面位置选择
	menu	生成用户输入选择菜单	printdlg	打印对话框
	msgbox	消息框	waitbar	显示等待条
	questdlg	问题对话框		
菜单设施	make-menu	建立菜单结构	umtoggle	改变用户界面菜单对象的检查状态
	menubar	为不同的计算机设定菜单条默认值	winmenu	为“Window”菜单项建立子菜单
工具条按钮群函数	btngroup	建立工具条按钮群	btnstate	工具条按钮群的排队状态
	btnpress	为工具条按钮群按动管理器	btndown	在工具条按钮群中按下按钮
	btnup	在工具条按钮群中抬起按钮		
用户定义	clruprop	清除用户定义特性	setuprop	设定用户定义特性
	getuprop	获取用户定义特性		

续表

杂项函数	allchild	获取所有子对象	popupstr	获取弹出菜单选择字符串
	findall	寻找所有对象	remapfig	变换图形对象的位置
	hidegui	隐藏/不隐藏 GUI	setptr	设定图形指针
	edtext	对轴系文本对象作交互的编辑	setstatus	设定图形中的状态文本字符串
	getstatus	获取图形中的状态文本字符串	overobj	获取指针过去后的对象句柄
	getptr	获取图形指针		

4.8　数据类型函数库（datatypes）

由于计算机应用十分广泛，要它处理的数据类型很多，仅以数为例，就有整数型（0～65 535，16 位），带符号整数型（－32 768～32 767，15 位加符号位），字符型（0～255，8 位），浮点单精度型（32 位），浮点双精度型（64 位）等，其他还有字符类型、指针类型等。在其他语言（例如 C 语言）中，编程时对每一个变量都要作出规定，这就增加了不少语句，而且要好好思考计划，不然就容易出错。

在 MATLAB 中，所有的数都用一种浮点双精度类型来存储和运算。因而省略了定义类型的语句，编程时无须去思考分辨，也减少了错误。当然对于那些本来只要用一两个字节来表示的变量来说，这种做法既浪费内存，又降低了运算速度。但用牺牲（存储）空间和（运算）时间来换取人机交互友善性的战略被证明是有效的，它形成了科学计算语言的特色，使人们不在编程的细节上费精力，而把注意力集中到科学计算的方法和建模合理性等大的问题上去。

在工程和管理系统中，常常需要分层次地把一些不同类型，不同尺寸和不同分层的数据组织起来，成为一个变量。MATLAB 专门为此定义了两种数据类型，一种称为结构阵列，另一种称为单元阵列。这两种数据类型在其他语言中也很少见，颇具特色，在此作一简单介绍。

4.8.1　结构阵列

假如我们要为一个班的学生建立一套管理档案 student，记录每个学生的三个项目：姓名（字符串）；出生日期（数字）、四门课（德育，数学，语文，体育）的成绩（数组）。这三项内容的数据类型各不相同，姓名的长度也不同，我们可以用在 student 后加域（field）的方法来建立一个结构阵列，键入：

```
student.name='John';
student.birthday='1985.06.15';
student.score=[85,78,92,68];
```

再键入 student，得到

```
ans = name: 'John'
```

```
        birthday: 1985.06
            score: [85.00 78.00 92.00 68.00]
```

也可用 struct 命令来建立结构阵列

```
student(2)=struct('name', 'Alice', 'birthday', '1986.01.20', 'score', ...
[77, 81, 65, 91]);
```

再键入 student，就得到

```
student =
1x2 struct array with fields:
        name
        birthday
        score
```

如果要得到 student 各个分量的详细内容，可键入

```
for i=1: 2 disp(student(i)), end
```

也可以提取各个域的内容，如键入

```
student.score
```

得到

```
ans = 85.00      78.00      92.00      68.00
ans = 77.00      81.00      65.00      91.00
```

结构阵列也可以嵌套。例如，若有的分数用优、良、中等级打，则也必须把课程成绩用结构阵列来表示成为 student.score.molarity，student.score.math，student.score.lang 等。

4.8.2 单元阵列

单元阵列是用来分层次地组织不同类型数据成为一个变量的另一种方法，与“结构阵列”的不同在于它不用域名，而是像矩阵那样用下标，与数值或字符串矩阵的区别在于用花括号代替方括号。例如上述 student 也可用单元阵列来存储，键入：

```
name={'John'; 'Alice'};
birthday={'1985.06.15'; '1986.01.20'};
score={[85, 78, 92, 68]; [77, 81, 65, 91]};
student={name; birthday; score};
```

再键入 student，得到

```
student = {2x1 cell}
        {2x1 cell}
            {2x1 cell}
```

键入 student{1}，得到

```
ans= 'John'
'Alice'
```

键入 student{1}{2}，得到

```
ans='Alice'
```

键入 student{3}{2}，得到

ans = 77 81 65 91

键入 student{3}{2}(2：3)，得到

ans = 81 65

最后的一个是圆括号，因为到这一级是按数组定义的，而前两极都是按单元阵列定义的。

同一个例子，可以用不同数据类型，也可以用不同的分级方式：

(1) student————————————姓名————————————人

————生日

————成绩

(2) student————————————人————————————姓名

————生日

————成绩

其对应的分级构造如图 4-14 所示。

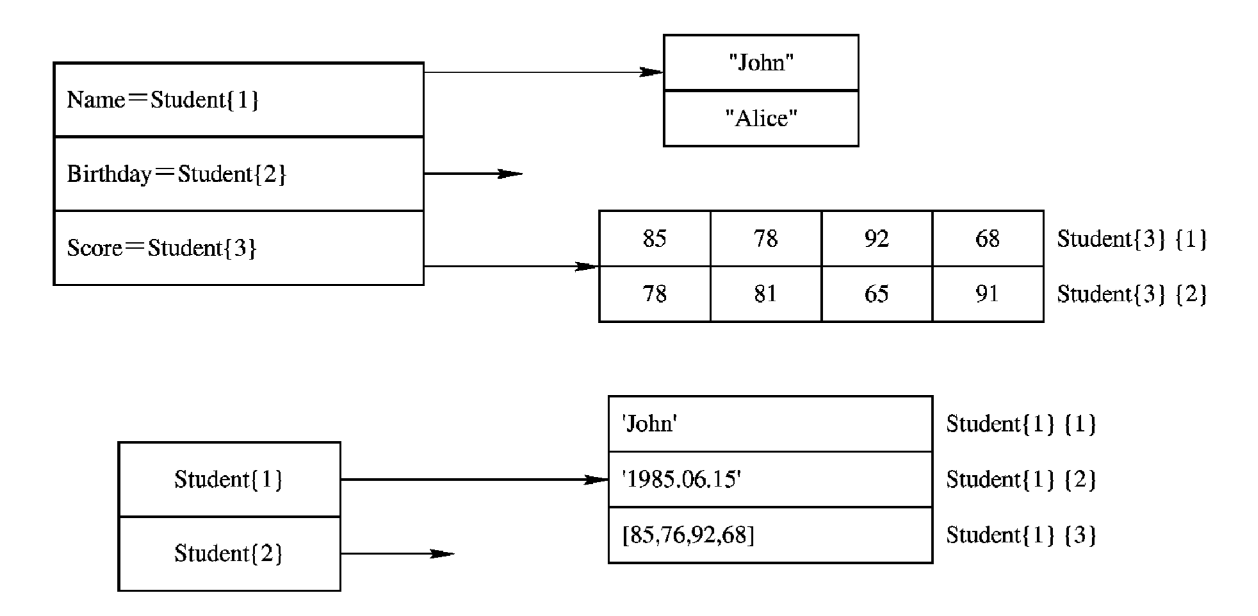


图 4-14 两种不同的数据构造方法

前一种方法在输入新人的时候不大方便。如果要把它改成后一种，即使得：

student{1}{1}='John'

student{1}{2}= '1985.06.15'

student{1}{3}= 85 78 92 68

那就应该按下述语句输入：

student{1}={'John'；'1985.06.15'；[85，78，92，68]}

student{2}={'Alice'；'1986.01.20'；[77，81，65，91]}

再键入 student，得

student = {3x1 cell} {3x1 cell}

即变量 student 由两个 3x1 的单元阵列组成，该单元阵列的三列分别为姓名、生日和成绩。

而键入 student{1}，得

```
ans =   'John'
        '1985.06.15'
        [1x4 double]
```

由此可见，结构阵列和单元阵列具有相似的功能，其差别在于使用的习惯不同。结构阵列利用域名来存储和调用各个元素，而单元阵列则是利用下标。前者的好处是具有明确的意义，便于记忆；而后者则比较简洁。

1. 类和对象

所谓“类”就是包含了对一种特定的变量结构及其运算方法的定义。所谓“对象”就是指一种特定“类”中的某个变量或某个实例。所谓“面向对象的编程”就是描述如何利用“类”和对“对象”进行编程的方法。

MATLAB 的基本部分有五种固有(built-in，已编好的)的类：

Double　　浮点双精度类

Sparse　　稀疏矩阵类

Char　　字符串阵列类(char，用双字节存储)

Struct　　结构阵列类

Cell　　单元阵列类

随着应用领域的扩大，某些 MATLAB 工具箱中也增加了一些其他的数据类型，例如为图像处理用的 unit8 类型(用单字节存储)，符号推理工具箱提供了 sym 类等。结构阵列和单元阵列的应用大大扩展了变量的类型。例如上述的变量 student 就可以看作一种新的变量类型。

这些数据类的运算规则和原来为双精度类型设计的 MATLAB 命令及函数当然不同，所以它们原则上不能使用＋、－、*、/、　等运算符和其他许多函数。MATLAB 采取了扩展运算符的方法，就是在采用一种特定的数据类型时，可为该种类型专门定义相应的运算符，例如用 plus 代替＋，mtimes 代替 * 等(参见表 4－11 的最后部分)，这些函数对不同的数据类型各不相同，因此它们被存储在不同数据类型的子库中。比如键入

```
help plus
```

在对 plus 函数作出说明以后，最后显示：

```
Overloaded methods
help polynom/plus.m
help sym/plus.m
help zpk/plus.m
help tf/plus.m
help ss/plus.m
```

表示系统中对 polynom，sym，zpk，tf，ss 这五种数据类型分别有 plus.m 文件定义了 plus 运算符。其中 zpk，tf，ss 三种数据类型用于控制系统工具箱中，后面将会介绍。polynom 和 sym 两种数据类型则分别用于多项式和符号运算中。在 4.3 节中已经看到，多项式的加法不能简单用矩阵加法，而要单独编一个子程序 polyadd 来完成。多项式的乘法也不能简单用矩阵乘法，而要用 conv 来实现，如果我们把 polyadd 和 conv 放在 polynom 类的方法库中，并分别命名为 plus 和 mtimes，那么就可以用“＋”和“*”来对多项式做运算了。

MATLAB 在执行程序时，将先对被运算的变量进行检验，如果确认其属于 polynom 类别，就会调用其方法库中的运算扩展符函数 polyadd 和 conv 来执行“＋”和“＊”运算符。特定的变量类型加上专门对这种变量类型进行运算的函数，就构成一个新的类。有关“类”和“对象”的概念，在初入门和基本应用中还不必用，在控制系统工具箱中将会用到，本书不予讨论。

表 4－11 给出了数据类型和结构函数库。

表 4－11　数据类型和结构(datatypes)(b)

数据类型	double	变换为双精度	sparse	建立稀疏矩直接
	char	建立字符(数组)串	cell	建立单元阵列
	struct	建立或变换为结构阵列	uint8	变换为无符号 8 位整数
	inline	构成内联(INLINE)对象		
多维数组函数	cat	链接数组	ndims	维数
	ndgrid	生成 N 维函数的阵列并插值	permute	重新排列矩阵维数
	ipermute	反向重排矩阵维数	shiftdim	移动维数
	squeeze	去除单项维数(降维)		
单元阵列函数	celldisp	显示阵列内容	cellplot	显示阵列的图形描述
	num2cell	把数字数组变换为单元阵列	deal	把输入分配给输出
	cell2struct	把单元阵列变换为结构阵列	struct2cell	把结构阵列变换为单元阵列
	iscell	是单元阵列时为真		
结构函数	struct	建立或变换为结构阵列	fieldnames	获取结构域名
	getfield	获取结构域的内容	setfield	设定结构域的内容
	rmfield	去除结构域	isfield	若域在结构阵列中时为真
	isstruct	是结构时为真		
面向对象编程函数	class	建立对象或返回对象类	struct	把对象变换为结构类
	methods	显示类的方法名	isa	若对象是给定类时为真
	isobject	是对象时为真	inferiorto	下级类关系
	superiorto	上级类关系		
可扩展的运算符	minus	扩展的 a～b	plus	扩展的 a＋b
	times	扩展的 a.＊b	mtimes	扩展的 a＊b
	mldivide	扩展的 a　b	mrdivide	扩展的 a/b
	rdivide	扩展的 a./b	ldivide	扩展的 a.　b
	power	扩展的 a.^b	mpower	扩展的 a^b
	uminus	扩展的 ～～a	uplus	扩展的 ＋a
	horzcat	扩展的 [a b]	vertcat	扩展的 [a; b]
	le	扩展的 a<＝b	lt	扩展的 a<b
	gt	扩展的 a>b	ge	扩展的 a>＝b
	eq	扩展的 a＝＝b	ne	扩展的 a～＝b
	not	扩展的 ～a	and	扩展的 a&b
	or	扩展的 a|b	subsasgn	扩展的 a(i)＝b，a{i}＝b 和域＝b
	subsref	扩展的 a(i)，a{i}和一个域	colon	扩展的 a:b
	transpose	扩展的 a′.	ctranspose	扩展的 a′
	subsindex	扩展的 x(a)		

语言篇作业

1. 求下列联立方程的解

$$\begin{aligned} 3x+4y-7z-12w &= 4 \\ 5x-7y+4z+2w &= -3 \\ x+8z-5w &= 9 \\ -6x+5y-2z+10w &= -8 \end{aligned}$$

2. 设 $A=\begin{bmatrix} 1 & 4 & 8 & 13 \\ -3 & 6 & -5 & -9 \\ 2 & -7 & -12 & -8 \end{bmatrix}$　$B=\begin{bmatrix} 5 & 4 & 3 & -2 \\ 6 & -2 & 3 & -8 \\ -1 & 3 & -9 & 7 \end{bmatrix}$

求 C1＝A * B′；C2＝A′ * B；C3＝A. * B，并求它们的逆阵。

3. (a) 列出 4×4 阶的单位矩阵 I，魔方矩阵 M 和 2×4 阶的全幺矩阵 A，全零矩阵；

(b) 将这些矩阵拼接为 8×8 阶的矩阵 C：

$$C=\begin{bmatrix} I & A'B' \\ A & \\ B & M \end{bmatrix}$$

(c) 求出 C 的第 2，4，6，8 行，组成 4×8 阶的矩阵 C1，及第 2，4，6，8 裂，组成 8×4 阶的矩阵 C2；

(d) 求 D＝C1 * C2 及 D1＝C2 * C1。

4. 设 $y=\cos x\left[0.5+\dfrac{3\sin x}{(1+x^2)}\right]$，把 x＝0～2π 间分为 101 个点，画出以 x 为横坐标，y 为纵坐标的曲线。

5. 求代数方程 $3x^5+4x^4+7x^3+2x^2+9x+12=0$ 的所有根。

6. 把 1 开五次方，并求其全部五个根。(提示：解 $x^5-1=0$)

7. 设方程的根为 x＝[－3，－5，－8，－9]，求它们对应的 x 多项式的系数。

8. 设微分方程

$$\frac{d^4y}{dt^4}+2\frac{d^3y}{dt^3}+5\frac{d^2y}{dt^2}+4\frac{dy}{dt}+3=u$$

求输入 u(t)＝δ(t)时的输出 y(t)。

9. 产生 8×6 阶的正态分布随机数矩阵 R1，求其各列的平均值和均方差。并求全体的平均值和均方差。

10. 产生 4×6 阶的均匀分布随机数矩阵 R，要求其元素在 1～16 之间取整数值。并求此矩阵前四列组成的方阵的逆阵。

11. x＝r cost＋3t，y＝r sint＋3，分别令 r＝2，3，4，画出参数 t＝0～10 区间生成的 x～y 曲线。

12. x＝sin t，y＝sin(Nt＋α)，

(a) 若 α＝常数，令 N ＝ 1，2，3，4，在四个子图中分别画出其曲线；

(b) 若 N=2，取 $\alpha=0$，$\pi/3$，$\pi/2$，及 π，在四个子图中分别画出其曲线。

13．设 $f(x)=x^5-4x^4+3x^2-2x+6$

(a) x=[-2，8]之间函数的值(取 100 个点)，画出曲线，看它有几个过零点；(提示：用 polyval 函数)

(b) 用 roots 函数求此多项式的根。

14．设 x=z sin3z，y=z cos3z，要求在 z=0～10 区间内画出 x，y，z 三维曲线。

15．设 $z=x^2e^{-(x^2+y^2)}$，求在定义域 x=[-2，2]，y=[-2，2]内的 z 值(网格取 0.1 见方)。

16．设 z1=0.05x-0.05y+0.1；画出 z1 的曲面(平面)图，迭合在上题的图中。

17．如 16 题。求出两曲面的交线，以叉号在图中标出。

18．将 13 题写成一个函数文件 f1.m，用 fzero 函数求它的过零点。与 13 题的结果进行比较讨论。如果在该函数中加一项 x sin(x)，过零点该怎样求？

19．设 $f(x)=\dfrac{1}{(x-2)^2+0.1}-\dfrac{1}{(x-3)^4+0.02}$，写出一个 MATLAB 函数程序 f31.m，使得调用 f1 时，x 可用矩阵代入，得出的 f(x)为同阶矩阵。画出 x=[0，4]区间内的 f31 曲线。

20．设 $f(x)=x^3-2x^2\sin x+5x\cos x+\dfrac{1}{x}$

(a) 画出它在 x=[0，4] 区间内的曲线，求出它的过零点的值；

(b) 求此曲线在 x 轴上方及下方的第一块所围的面积的大小。

21．知微分方程：$\dfrac{dy}{dx}=\dfrac{x^2}{y}-x\cos y$，若 y(0)=1，求它在 x=[0，5]区间内的数值积分，并画出曲线。

22．eval 命令执行字符串 s='y=magic(3)'。

23．如果要用 for 循环及 eval 语句实现 yn=magic(n)，(n=3，4，5)，请编出程序。

24．用 sprintf 命令写出字符串'自然对数底数 e=2.71828……'，e 的值应该由 MATLAB 自动生成，其小数点后要显示 20 位。

25．用字符串、单元阵列及结构阵列三种方式定义 student1，student2 及 student3 三个数组，此数组应包括 John，David 及 Tom 三个人名。请比较三者的不同。如果还要包括第二属性——他们的出生地 birthplace，分别为 Shanghai，Nanjing 和 Hangzhou，又有什么差别？

20．方程① $x^3+\cos(a)=0$，② $x^3+\cos(x)=0$ 及③ $x^3+\cos(ax)=0$，用符号运算工具箱函数 solve 分别求 x 的解。用 a=0.5 代入，求 x 的数值解，并与用 roots 函数所求的结果进行比较。

第二篇　应　用　篇

本篇介绍 MATLAB 在大学本科(以电气工程和机械工程专业为典型)课程中的应用，主要涉及基础课和部分专业基础课，包括高等数学、普通物理、力学、机械、电工电路和信号系统等。有一些重要的专业基础课，像自控原理、信号处理等，已经有专门著作，并且要用到工具箱，本书就不再涉及。

作为一种优良的科学计算工具，MATLAB 对于所有需要较多计算和作图的课程，都有很大的应用价值。大学生从一年级到四年级，如果能在各门课中用同一种算法语言，那不仅可以比较熟练地掌握该语言，而且可以充分利用各门课中数学计算的共性，用同样的子程序解决不同课程的问题，从而大大提高了工作效率。

本篇由涉及十多门课的大量例题组成，对于每门课而言，一般不超过 10 个例子。这些例子当然无法完全覆盖这些课的所有内容，我们的目的是使读者通过这些例题，掌握 MATLAB 语言中各种命令的用法，并且了解在各门课中如何入门，从而写出自己的解题程序。

为了压缩篇幅，又能列入较多的例题，我们把 MATLAB 程序中的许多输入、输出和图形标注语句省略了。因为这些语句占很大篇幅，且大同小异，本篇重点介绍各题中的核心程序。为了使读者能方便地运行这些程序，我们的网站提供了完整的免费下载程序。

第 5 章 在高等数学中的应用举例

5.1 函数、极限和导数

一、单变量函数的计算和绘图

【例 5-1-1】 单变量函数的计算和绘图，设

$$y=\frac{\sqrt{3}}{2}\cdot e^{-4t}\sin\left(4\sqrt{3}t+\frac{\pi}{3}\right)$$

要求以 0.01 s 为间隔，求出 y 的 151 个点，并求出其导数的值和曲线。

解：

◆建模

可以采取以下两种方法：

(1) 直接用程序文件编程的方法。

(2) 编成函数文件，由主程序调用的方法。

求导数采用 diff 函数。

◆MATLAB 程序

(1) 第一种方法的程序如下(ex511a)：

```
t=[0: .01: 1.5];             % 设定自变量数组 t
w= 4 * sqrt(3);              % 固定频率
y=sqrt(3)/2 * exp(-4 * t). * sin(w * t + pi/3);    % 注意用数组运算式
subplot(2, 1, 1), plot(t, y), grid       % 绘制曲线并加上坐标网格
title('绘图示例'), xlabel('时间 t'), ylabel('y(t)')    % 加标注
% 求导数并绘制导数曲线，注意数组求导数后其长度减少 1
Dy=diff(y); subplot(2, 1, 2), plot(t(length(t)-1), Dy), grid
ylabel('Dy(t)')        % 加标注
```

(2) 第二种方法，其主程序为(ex511b)：

```
dt=0.01; t=[0: dt: 1.5]; w= 4 * sqrt(3);
y=ex511bf(t, w); Dy=diff(y)/dt;
% 绘图和加标注的程序略去
```

另要建立一个函数文件 ex511bf.m，其内容为

```
function xvalues= ex511bf (tvalues, w)
```

```
% ex511bf 是例 5-1-1 用的函数文件，它应该符合元素群运算规则
xvalues=sqrt(3)/2*exp(-4*tvalues).*sin(w*tvalues + pi/3);
```

◆程序运行结果

运行这两个程序都得到图 5-1-1 所示的曲线。为了节省篇幅，我们没有显示 y 的数据。以后的各例中还将删略绘图时的标注语句。从本例看，第二种方法似乎更麻烦一些，但它具备模块化的特点。当程序中要反复多次调用此函数，而且输入不同自变量时，利用函数文件可大大简化编程。我们应该掌握这种方法。

两次应用 diff 函数或用 diff(y, 2)，除以两次 dt，可以求 y 的二次导数，读者可自行实践。

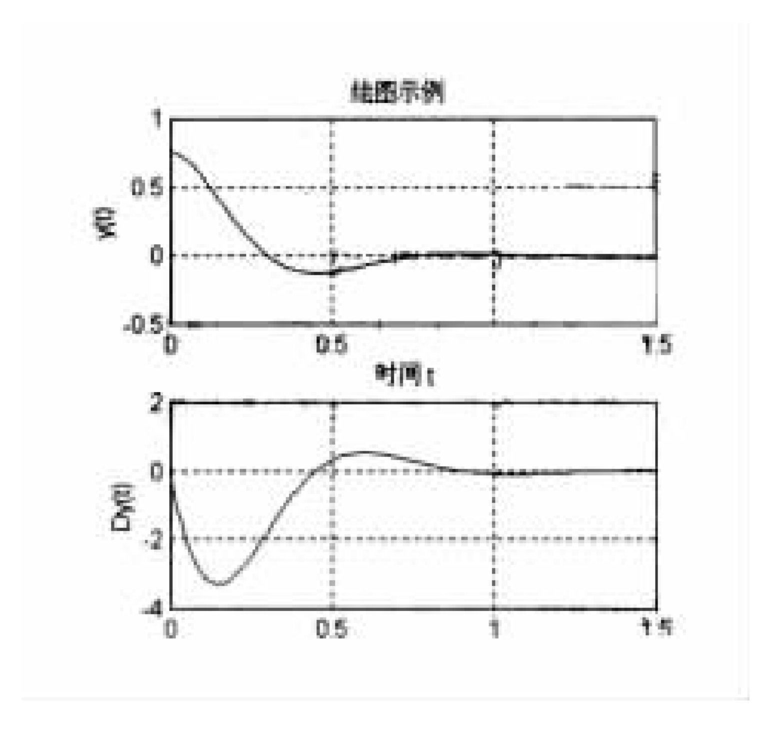


图 5-1 例 5-1-1 的曲线

【例 5-1-2】 绘制极坐标系下曲线

$$\rho = a\cos(b + n\theta)$$

并讨论参数 a、b、n 的影响。

解：

◆建模

为了便于比较，编一个能分别画出两个图形的程序，采用 for 循环，读者可从中看到利用循环指数的技巧。

◆MTALB 程序

```
theta=0:0.1:2*pi;          % 产生极角向量
for i=1:2
    a(i)=input('a='); b(i)=input('b='); n(i)=input('n=')
    rho(i,:)=a(i)*cos(b(i)+n(i)*theta);       % 极坐标方程
    subplot(1,2,i), polar(theta, rho(i,:));      % 极坐标系绘图
end
```

◆程序运行结果

运行并输入不同参数的结果如图 5－2。

a＝2；b＝pi/4；c＝2　　　　(4 叶玫瑰线)

a＝2；b＝0；c＝3　　　　(3 叶玫瑰线)

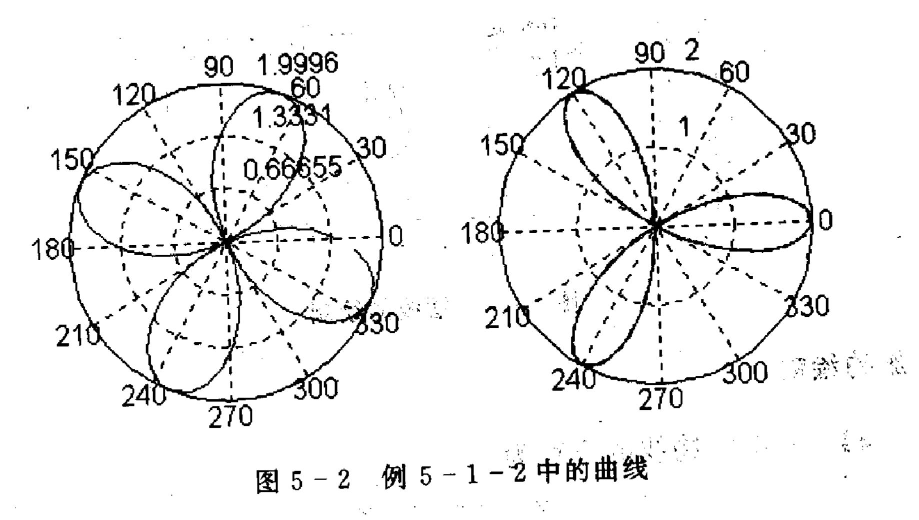


图 5－2　例 5－1－2 中的曲线

二、参变方程表示的函数的计算和绘图

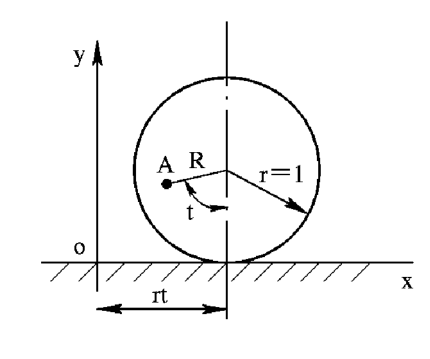


图 5－3　摆线生成

【例 5－1－3】　摆线的绘制。当圆轮在平面上滚动时，其圆周上任一点所画出的轨迹称为摆线(如图 5－3)所示。如果这一点不在圆周上而在圆内，则生成内摆线；如果该点在圆外，离圆心距离大于半径，则生成外摆线。后一种情况，可想象成火车轮，其接触轨道的部分，并不是直径最大处，而内侧的直径还要大一些，以防止车轮左右出轨，在这部分边缘上的点就画出外摆线。

解：

◆建模

概括几种情况，其普遍方程可表示为(设 r 为轮半径，R 为点半径)

$$x_A = rt - R\sin t$$

$$y_A = R\cos t$$

可由这组以 t 为参数的方程分析其轨迹。

◆MATLAB 程序

```
t=0:0.1:10;
r=input('r='),
R=input('R=')
x=r*t-R*sin(t);y=r-R*cos(t);
plot(x,y),axis('equal')
```

◆程序运行结果

设 r=1，令 R=r 时得到摆线，当 R=0.7 及 R=1.5 时得到内摆线和外摆线，这 3 根摆线均绘于图 5－4 中。为了显示摆线的正确形状，x，y 坐标保持等比例是很重要的，因此程序中要加 axis('equal')语句。

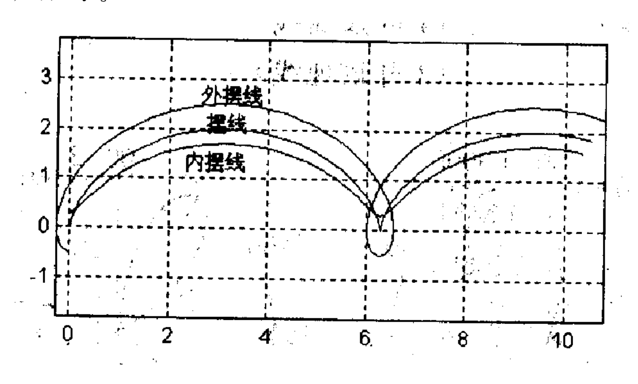


图 5－4 摆线的绘制

三、曲线族的绘制

【例 5－1－4】 三次抛物线的方程为

$$y = ax^3 + cx$$

试探讨参数 a 和 c 对其图形的影响。

解：

◆建模

因为函数比较简单，可以直接写入绘图语句中，用循环语句来改变参数。注意坐标的设定方法，以得到适于观察的图形。给出的程序不是惟一的，例如可用 fplot 函数编程，读者可自行探索其它编法。

◆MATLAB 程序

```
x=－2：0.1：2；            % 给定 x 数组，确定范围及取点密度
subplot(1，2，1)           % 分两个画面绘图
for c=－3：3  plot(x，x.^3＋c＊x)，hold on，end，grid    % a=1，取不同的 c
axis('equal')，axis([－2  2  －3  3])    % x，y 坐标等比例并确定其范围
subplot(1，2，2)，
for a=－3：3  plot(x，a＊x.^3＋x)，hold on，end，grid    % c=1，取不同的 a
axis('equal')，axis([－2，2，－3，3])
% 用 gtext 命令在图内标注字符
```

◆程序运行结果

运行结果如图 5－5 所示。其中 a 和 c 均从－3 取到 3，步长为 1。

四、极限判别

【例 5－1－5】 极限的定义和判别。

用 MATLAB 语言来表达推理过程是比较困难的，它必须与实际的数值联系起来，比如无法用无穷小和高阶无穷小的概念，只能用 10^{-10}，10^{-20} 等数值。极限的定义恰恰是用了 δ 和 ε 等数值的概念，因此不难用程序表达。

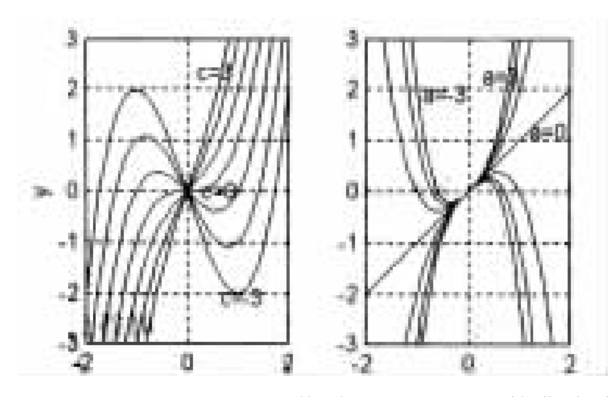


图 5-5　c 和 a 取不同值时 $y=ax^3+cx$ 的曲线族

解：

◆建模

用函数极限的定义，对于函数 $y=f(x)$，当任意给定一个正数 ε 时，有一个对应的正数 δ 存在，使得

$$0<|x_c-x|<\delta \text{ 时}，|A-f(x)|<\varepsilon$$

则 A 就是 f(x)在 $x\to x_c$ 时的极限，如果找不到这样的 δ，A 就不是它的极限。只考虑左极限时，因 x_c-x 必为正数，可去掉绝对值符号。

◆检验左极限是否正确的程序

```
disp('A 是否是 f(xc)的左极限?')
A=input('A=，例如 A=1')，                    % 输入极限值
xc=input('xc=，例如 xc=0')，                 % 输入对应的自变量值
fxc=input(' f(x)的表达式为，例如 sin(x)/x'，'s')，      % 输入函数表达式
flag=1；delta=1；x=xc-delta；n=1；                     % 初始化
while flag==1 epsilon=input('任给一个小的数 ε=')       % 任意给出 ε
 while abs(A-eval(fxc))>epsilon   delta=delta/2；x=xc-delta；    % 找 δ
      if abs(delta)< eps disp('找不到 δ')，n=0；break   % 找不到 δ，跳出内循环
      end，end
    if n==0    disp('左极限不正确')，break，end，    % 极限不正确，跳出外循环
    disp('δ=')，delta                               % 找到了 δ
    disp('左极限可能正确')
 flag=input('再试一个 ε 吗？再试按 1，不试按 0 或任意数字键 ')，   % 是否再试
end
```

◆程序运行结果

我们来检验：

(1) $f(x)=x^2-8$ 在 $x\to x_c=3$ 时是否以 1.001 为左极限。

(2) $f(x)=\dfrac{\sin x}{x}$在 $x\to x_c=0$ 时是否以 1 为左极限。

对(1)得出的结果将是“左极限不正确”，而对(2)将得出“左极限可能正确”的结果。读者可分析为什么要加“可能”二字而不给出肯定答复。

5.2 空间解析几何

一、曲面的方程

【例 5-2-1】 二次曲面的方程如下

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}=d$$

要求讨论参数 a，b，c 对其形状的影响，并画出其图形。

解：

◆建模

本题的数学模型很清楚，关键在于如何作出三维曲面图形。特别要注意在给定了 x，y 值求 z 时，若有开方运算，一是会出现虚数，二是对实数也有正负两个解。为了使虚数不出现在绘图中，采用了一种技巧，那就是把虚数都换成非数(NaN)。

◆MATLAB 程序

```
a=input('a='); b=input('b='); c=input('c=');
  d=input('d='); N=input('N=');              % 输入参数，N 为网格线数目
  xgrid= linspace(-abs(a), abs(a), N);       % 建立 x 网格坐标
  ygrid= linspace(-abs(b), abs(b), N);       % 建立 y 网格坐标
  [x, y]=meshgrid(xgrid, ygrid)          % 确定 N×N 个点的 x, y 网格坐标
  z=c * sqrt(d-y. * y/b/b-x. * x/a/a); u=1;     % u=1，表示 z 要取正负值
  z1=real(z);                    % 取 z 的实部 z1
  for k=2: N-1                  % 以下 7 行程序的作用是取消 z 中含虚数的点
      for j=2: N-1
        if imag(z(k, j))~=0     z1(k, j)=0; end
        if all(imag(z([k-1: k+1], [j-1: j+1])))~=0     z1(k, j)=NaN;
        end
      end
  end
  surf(x, y, z1), hold                       % 画空间曲面
  if u==1 z2=-z1; surf(x, y, z2);           % u=1 时加画负半面，并加负坐标轴
      axis([-abs(a), abs(a), -abs(b), abs(b), -abs(c), abs(c)]);
  end
  xlabel('x'), ylabel('y'), zlabel('z')
  hold off
```

◆程序运行结果

运行结果如图 5-6 所示，特别注意 a，b 取虚数对图形的影响。

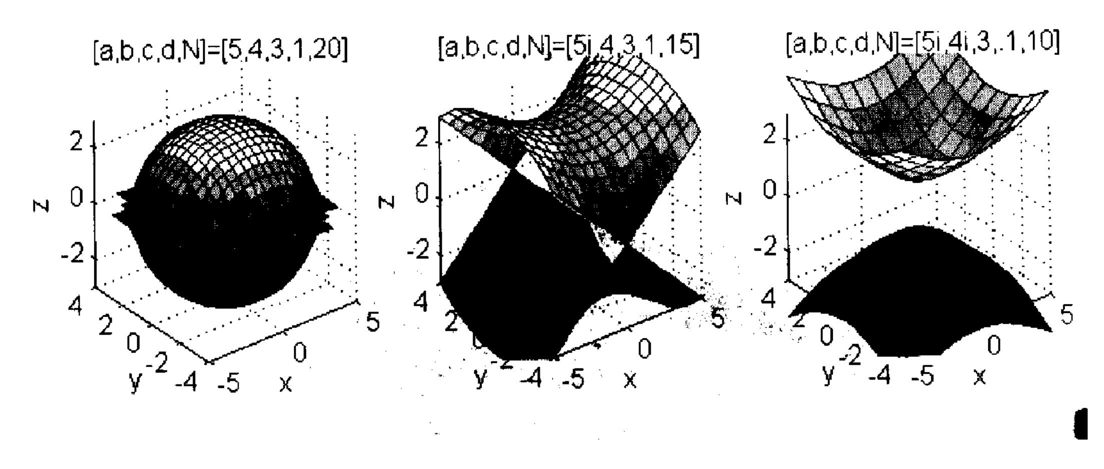


图 5－6 a，b，c，d，N 参数取不同值时所得不同形状的二次曲面

二、空间两曲面的交线

【例 5－2－2】 列出求空间两任意曲面的交线的 MATLAB 程序。

解：

◆建模

两空间曲面方程联立起来，就形成一个空间曲线的方程。这个曲线能满足两个曲面的方程，因而也就是这两个空间曲面的交线。显示这两个曲面不难，用两次 mesh 语句即可，但要显示其交线，必须先找到各个交点，因为数值计算得到的是离散点，难以找到两曲面上完全重合的点，本程序采用了设置门限的方法，只要在同一网格点处，两曲面的 z 值之差小于设定门限，就认为它是交点，门限值要试验几次才能定得好。

◆MATLAB 程序

```
%本程序给出两个空间曲面的交线(当然是空间曲线),给出不同的 z1,z2 方程可画出不同的空间曲面和其交线
[x, y]=meshgrid(-2: .1: 2);           % 确定计算和绘图的定义域网格
z1=x. * x-2 * y. * y;                  % 第一个曲面方程
z2=2 * x-3 * y;                        % 第二个曲面方程(平面)
mesh(x, y, z1); hold; mesh(x, y, z2);      % 在一个图上同时画出两个曲面
r0=(abs(z1-z2)<=.1);                   % 求两曲面 z 坐标差小于 0.1 的网格矩阵
zz=r0. * z1; yy=r0. * y; xx=r0. * x;    % 求这些网格上的坐标值，即交线坐标
plot3(xx(r0~=0), yy(r0~=0), zz(r0~=0), '*');       % 画出这些点
colormap(gray), hold off             % 不用彩色而用灰度表示曲面
```

◆程序运行结果

执行此程序得出的曲面如图 5－7 所示。

如果想改变曲面方程，可以在程序中改动第二行和第三行。但这样的程序还不是通用的。最好程序运行时能向用户提问，允许用户输入曲面方程。此时就要用到字符串功能和 eval 命令

```
s1=input('输入第一个方程', 's');
```

在原来的 z1 方程语句处改为

```
z1 = eval(s1);
```

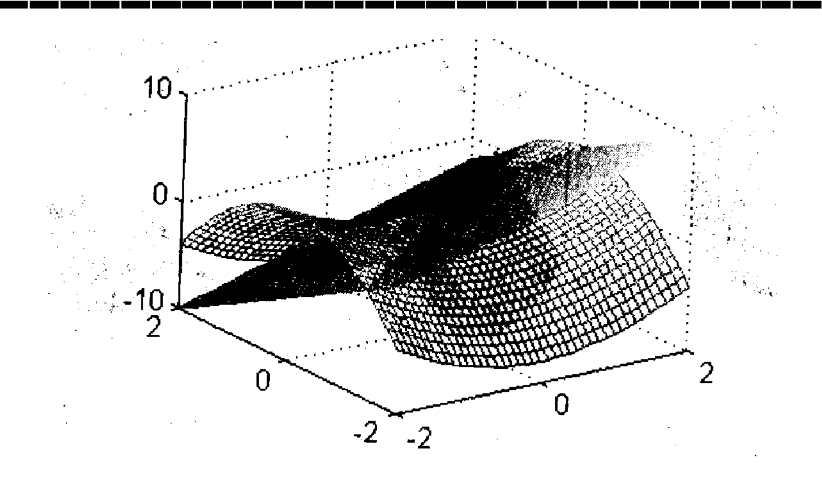


图 5－7 两曲面交线

类似地输入第二个方程。

此外，应使用户能给出定义域和间隔。这实现起来比较简单，只要把第一句改为

[x，y]＝meshgrid(xmin：dx：xmax，ymin：dy：ymax)；

其中，xmin，dx，xmax，ymin，dy，ymax 可由程序给出屏幕提问，让用户用键盘输入。当然，这样又增加了运行时的麻烦，所以编程时要找一个折衷的选择，要有一定的灵活性又不能太麻烦，应恰到好处。

【例 5－2－3】 用平行截面法讨论由方程 $z=x^2-2y^2$ 构成的马鞍面形状。

解：

◆建模

对上例的程序做如下修改：

① 定义域网格改为

[x，y] ＝ meshgrid(－10：.2：10)

② 第一个曲面方程改为

z1 ＝ (x.^2 － 2 * y.^2) ＋ eps

③ 第二个曲面(平面)方程改为与 z 轴正交的水平面

z2 ＝ a

④ 为了画 z2 的曲面图，应使 z2 与 x，y 有同样的维数，故写成

z2 ＝ a * ones(size(x))

a 可由用户输入，另外用 subplot 把曲面和交线分别画在两张图上，并注意把两个分图取成同样比例，便于比较，因为 z 的范围增大，必须把两曲面交点处 z1 和 z2 的容差放大到 1。

◆MATLAB 程序

```
[x，y]＝meshgrid(－10：.2：10)；        % 确定计算和绘图的定义域网格
z1＝(x.^2－2 * y.^2)＋eps；        % 第一个曲面方程
a＝input('a＝(－50<a<50)')；z2＝a * ones(size(x))；        % 第二个方程(平面)
subplot(1，2，1)，mesh(x，y，z1)；hold on；mesh(x，y，z2)；% 分别画出两个曲面
v＝[－10,10,－10,10,－100,100]；axis(v)，grid        % 确定第一个分图的坐标系
colormap(gray)，hold off，        % 取消彩色，改为灰度
```

```
r0=abs(z1-z2)<=1;            % 求两曲面 z 坐标差小于 1 的网格
zz=r0.*z2; yy=r0.*y; xx=r0.*x;   % 求这些网格上的坐标值，即交线坐标
subplot(1,2,2),plot3(xx(r0~=0),yy(r0~=0),zz(r0~=0),'x');% 画出交线
axis(v), grid                % 使第二个分图取第一个分图的坐标系
```

设计 MATLAB 程序时，要注意避免本程序对后续程序的影响，因此，凡是用过 hold 语句的程序，最好结尾加一个 hold off；用过 subplot 语句的，最好加一个 subplot(1，1，1)来复原。但执行 subplot(1，1，1)以后，原有的图形会消失。因此末句应为

```
pause, subplot(1, 1, 1)
```

为了避免先行程序中这类图形设定语句对本程序的影响，程序的首行最好以 clf 开始。

◆程序运行结果

执行此程序，并输入 a=8，所得三维图形见图 5-8(*a*)，截面与曲面的交线见图 5-8(*b*)。输入不同的 a 就可得不同的横切面形状。图 5-8(*b*)中下面一对曲线是设 a=-20 时得到的，可见从上而下，其横切面交线发生了很大的变化。

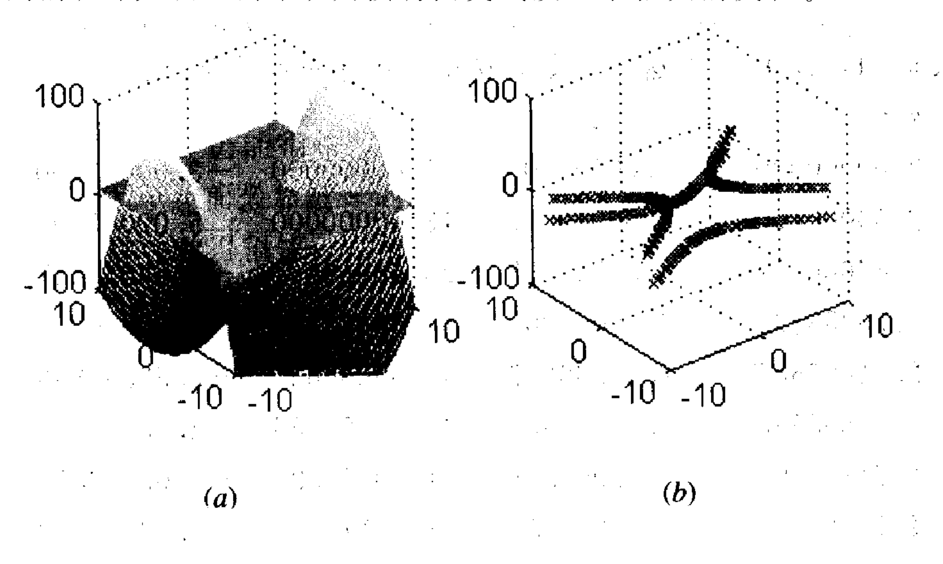


图 5-8　鞍形曲面的水平截面

5.3　数 列 和 级 数

【例 5-3-1】　数列的表示方法。

解：

◆建模及程序

MATLAB 中的元素群运算特别适合于简明地表达数列，它可省去其它语言中的循环语句。下面就是数列$\frac{1}{n}$，$\frac{(-1)^n}{n}$，$\frac{1}{n(n+1)}$在 n=1～6 时的 MATLAB 表示式

```
n=1:6;
1./n=            1.0000   0.5000    0.3333   0.2500    0.2000  0.1667
```

(-1).^n./n= -1.0000 0.5000 -0.3333 0.2500 -0.2000 0.1667(交项级数)

1./n./(n+1)= 0.5000 0.1667 0.0833 0.0500 0.0333 0.0238

但在某些情况下，当数列的单项运算中包含元素群运算时，难免仍要用 for 循环来求整个数列，比如求数列$\frac{1}{n!}$，就不能直接用 1/prod(1: n)，因为 prod(1: n)是 n 的阶乘，它本身已是一个元素群运算，不能再让 n 作为数组，这时必须用

```
for i=1:6    x(i)=1/prod(1: i); end, x
```

得 x = 1.0000 0.5000 0.1667 0.0417 0.0083 0.0014

【例 5-3-2】 利用幂级数计算指数函数。

解：

◆建模及程序

指数可以展开为幂级数

$$e^x = 1 + x + x^2/2! + x^3/3! + \cdots + x^n/n! + \cdots$$

其通项为 x^n/prod(1:n)，因此用下列循环相加程序就可计算出这个级数

```
x=input('x='); n=input('n='); y=1;          % 输入原始数据，初始化 y
for i=1: n    y=y+x^i/prod(1: i); end, y     % 将通项循环相加 n 次，得 y
```

分别代入 x=1，2，4，-4 这 4 个数，取 n=10，y 的结果如下：

n	x= 1	2	4	-4
1	2.00000000	3.00000000	5.00000000	-3.00000000
2	2.50000000	5.00000000	13.00000000	5.00000000
3	2.66666667	6.33333333	23.66666667	-5.66666667
4	2.70833333	7.00000000	34.33333333	5.00000000
5	2.71666667	7.26666667	42.86666667	-3.53333333
6	2.71805556	7.35555556	48.55555556	2.15555556
7	2.71825397	7.38095238	51.80634921	-1.09523810
8	2.71827877	7.38730159	53.43174603	0.53015873
9	2.71828153	7.38871252	54.15414462	-0.19223986
10	2.71828180	7.38899471	54.44310406	0.09671958
有效数位(精度)	7	4	2	0

由此可以看出，这个简单程序虽然原理上正确，但不好用，对不同的 x，精度差别很大。其它存在的问题还有：

(1) 这个程序不能用于 x 的元素群运算。

(2) 当 x 为负数时，它成为交项级数，收敛很慢。

(3) 此程序要作 $n^2/2$ 次乘法，n 很大时，乘法次数太多，计算速度很低。

(4) 对不同的 x，要取不同的 n 才能达到精度要求。因此 n 不应由用户输入，应该由软件按精度要求来选。

◆程序改进的方法

针对这四个问题，可以采取下面四种方法。

(1) 允许数组输入，改进输出显示

```
x=input('x='); n=input('n='); y=ones(size(x));     % 输入 x，n，初始化 y
for i=1:n
    y=y+x.^i/prod(1:i);                             % 循环相加
    s1 = sprintf('%13.0f', i); s2 = sprintf('%15.8f', y); % 将结果变为字符串
    disp([s1, s2])                                  % 显示
end
```

执行此程序并输入 x=[1，2，4，−4]及 n=10，可一次得出上面的计算结果。

(2) 可以利用 exp(−x) = 1 / exp(x) 来避免交项级数的计算。

(3) 为了减少乘法次数，设一个中间变量 z，它的初始值为 z=ones(size(x))，把循环体中的计算语句改成

```
y = y + z; z = x. * z/i
```

这样求得的 z 就是 z=x.^i/i!，于是每个循环只需做一次乘法，计算整个级数只需 n 次乘法。按这种算法，y 的初始值应改为

```
y=zeros(size(x))
```

(4) 为了按精度选择循环次数，不该用 for 循环，而用 while 语句，它可以设置循环的条件语句，通常可取 y+z−y>tol，tol 是规定的允许误差。只要相邻两次的 y 值之差大于 tol，循环就继续进行，直至小于 tol 为止。

使 x 较大时，exp(x)仍能很快收敛，还可以利用关系式 $\exp(x)=(\exp(x/k))^k$，令 x1=x/k，k 通常取大于 x 而最靠近 x 的 2 的幂，例如 x=100，就取 k=128，以保证 x1 的绝对值小于 1，这时级数收敛得很快，从例中可以看出，n 取 10 时(即级数取 10 项)就能保证七位有效数。而 $\exp(x1)^{128}$ 可化成 $x = (\cdots((\exp(x1))^2)^2\cdots)^2$，即 exp(x1)的 7 次自乘，总共用 17 次乘法就可完成 $\exp(100)=(\cdots(\exp(100/128))^2\underbrace{\cdots)^2}_{7}$ 的计算，这既保证了精度，又提高了速度。

【例 5－3－3】 用泰勒级数表示一个多项式，显示级数取不同阶次对逼近程度的影响。

解：

◆建模

一个多项式函数可以精确地用泰勒公式展开，但必须取足够高的阶数(等于多项式的次数)，否则就会产生误差。在 MATLAB 中，多项式可以用其系数向量来表示，求值和求导用 polyval 和 polyder 命令(参阅 4.3 节)。

设 $f(x)=a_1x^n+a_2x^{n-1}+\cdots+a_nx+a_{n+1}$，则多项式在 $x=x_0$ 附近展开的泰勒公式为

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \cdots + \frac{f^{(n)}(x_0)}{n!}(x-x_0)^n$$

◆MATLAB 程序

此程序最高只到三阶，如果输入多项式系数向量 a 的长度不大于 4，则其高阶泰勒展开式完全精确。如果 length(a)大于 4，就会有误差。读者可试算并考虑如何编写更完美的程序。

```
a=input('输入多项式系数向量 a=[ ]=');          % 输入参数
x0=input('输入展开点的坐标值 x0= ');
[xm]=input('输入展开的坐标区间[xmin, xman]=');
x=linspace(xm(1), xm(2));                     % 设定自变量数组
y=polyval(a, x); ya=polyval(a, x0);   % 求 y 的准确值和 y 在 x0 点的值 y(x0)
Da=polyder(a), Dya=polyval(Da, x0); % 求 x0 点的一阶导数的值
D2a=polyder(Da), D2ya=polyval(D2a, x0);   % 求 x0 点的二阶导数的值
D3a=polyder(D2a), D3ya=polyval(D3a, x0);  % 求 x0 点的三阶导数的值
yt(1, :)=ya+Dya*(x-x0);                    % 求 y 的一阶泰勒级数
yt(2, :)=yt(1, :)+D2ya*(x-x0).^2/prod(1:2);   % 求 y 的二阶泰勒级数
yt(3, :)=yt(2, :)+D3ya*(x-x0).^3/prod(1:3);   % 求 y 的三阶泰勒级数
plot(x, y, '.', x, yt(1:3, :)), grid    % 绘图时准确值用点线表示
```

◆程序运行结果

输入 a=[2, -3, 4, 5];

x0=1; xmin=0; xmax=2;

得出图 5-9 所示的三根曲线，本来应有四根，但三阶泰勒级数和精确曲线是重合的，因为我们输入的多项式系数向量长度为 4。读者可试验输入更长的 a 来比较其结果。

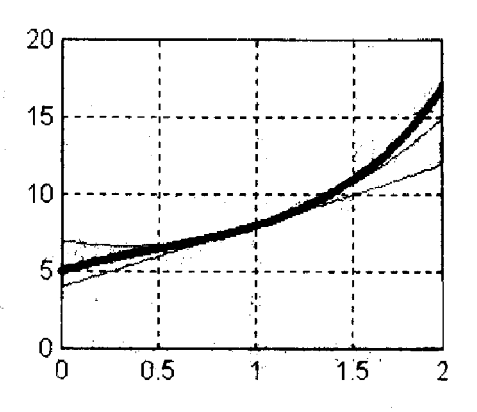


图 5-9 多项式的不同阶泰勒展开

【例 5-3-4】 编写任意函数展开为各阶泰勒级数的程序，并显示其误差曲线。

解：

◆建模

对任意函数 y=f(x)的泰勒展开式如下：

$$f(x)=f(a)+\frac{f'(a)}{1!}(x-a)+\frac{f''(a)}{2!}(x-a)^2+\cdots+\frac{f^{(n)}(a)}{n!}(x-a)^n+R_n(x)$$

其中，$R_n(x)$为余项，也就是泰勒级数展开的误差。

◆MATLAB 程序

```
fxs=input('输入 y=f(x)的表达式', 's');       % 输入原始条件，fxs 是字符串
K=input('输入泰勒级数的展开阶数 K= ');
a=input('展开的位置 a= '); b=input('展开的区间半宽度 b= ');
x=linspace(a-b, a+b);           % 构成自变量数组，确定其长度和步长
lx=length(x); dx=2*b/(lx-1);
y=eval(fxs);                                  % 求出 y 的准确值
subplot(1, 2, 1), plot(x, y, '.'), hold on  % y 的准确曲线用点线绘出
% 求出 a 点一阶导数，注意求导后数组长度减小 1
Dy=diff(y)/dx; Dya(1)=Dy(round((lx-1)/2));
yt(1, :)=y(round(lx/2))+Dya(1)*(x-a); % 求 y 的一阶泰勒展开，绘图
plot(x, yt(1, :))
```

```
for k=2: K
    Dy=diff(y, k)/(dx^k); Dya(k)=Dy(round((lx-k)/2)); % 求 a 点 k 阶导数
    yt(k,:)=yt(k-1,:)+Dya(k)/prod(1:k) * (x-a).^k; % 求出 y 的 k 阶级数
    plot(x, yt(k, :)),          % 绘图
    e(k, :)=y-yt(k, :);         % 求出 yt 的误差
end
title([fxs, '的各阶泰勒级数曲线']),          % 注意如何组成标注的字符串
grid, hold off, subplot(1, 2, 2)
for k=1: K   plot(x, e(k, :)), hold on, end      % 绘制误差曲线
title([fxs, '的各阶泰勒级数误差']), grid, hold off
```

◆程序运行结果

输入 fxs=cos(x)，K=5，a=0.5，b=2，所得曲线如图 5-10 所示。读者可改变其坐标系范围以仔细观察最关心的部分，也可输入其它函数作验算，注意输入函数应符合元素群运算规则。

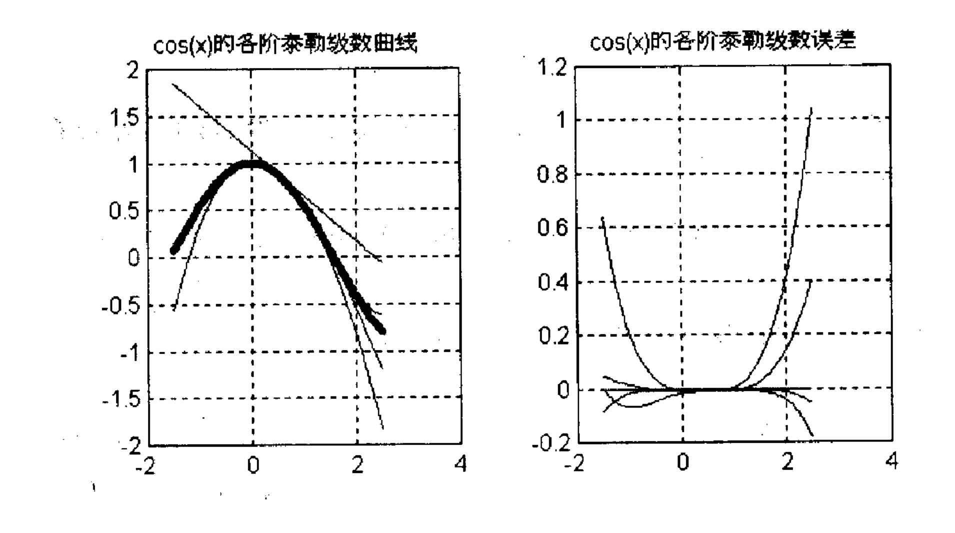


图 5-10　通用泰勒级数程序在 cos(0.5)附近展开的运行结果

5.4　数值方法和数值积分

一、任意非线性方程 f(x)=0 的解

这就是求任意曲线 y=f(x)过零点的问题。

【例 5-4-1】 用切线法求下列方程的近似数值解。

$$y = x^3 + 10x^2 - 2\sin x - 50$$

解：

◆建模

用 MATLAB 来解此题时，可先大致看一下曲线的形状。用 fplot 函数得出的曲线如图

5－11。它有 3 个过零点，分别在－10，－3，3 附近。函数的绘图范围要试探几次，先大一些，看到全部过零点，然后再缩小到上述能看清所有过零点的最小范围。

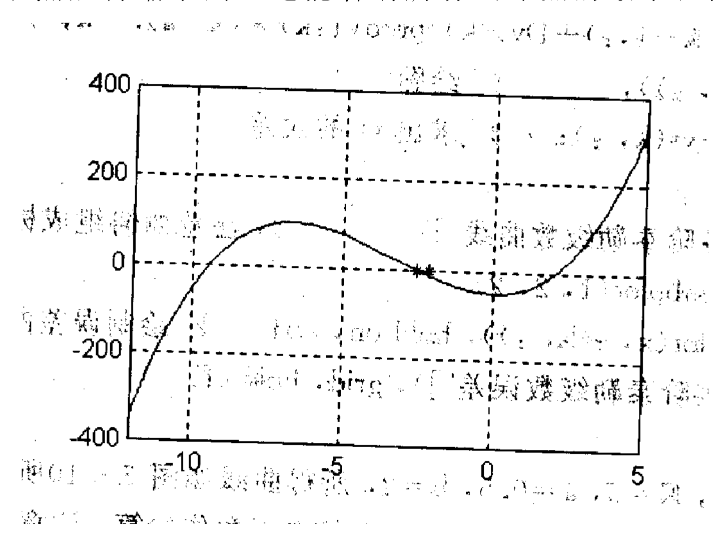


图 5－11 用切线法求方程的解

切线法求根是基于如下公式

$$x_x = x_0 - \frac{f(x_0)}{g(x_0)}$$

其中，x_0 为前一次的 x 试探值；x_x 为得出的新近似值；$g(x_0)$ 为 f(x)的导数在 x_0 点的值。

◆MATLAB 程序

```
 clf, fplot('y =x.^3+10*x.^2-2*sin(x)-50', [-12, 5]),
 hold on, grid
x0=input(' x0=给出要求的解的近似坐标') ; e=1
while e>0.0001
    f= x0.^3+10*x0.^2-2*sin(x0)-50;          % 求 f(x0)
    g=3*x0^2+20*x0-2*cos(x0);             % 求函数的导数 g(x0)
    xx=x0-f/g; e=abs(xx-x0) ;              % 切线法求解公式，精度计算
    x0=xx ;                                % 把新值赋予 x0

    plot(x0, 0, '*'), hold on, pause(1)    % 画出新点在 x 轴上的位置
end
```

◆程序运行结果

运行此程序，设 x0 分别为－10，－2，2，得出的解分别为 －9.4384，－2.5648，2.0707，并可从图上看到解逐渐接近精确解的过程。本程序中用了导数的解析形式，对某些较为复杂的函数，这会很繁琐，所以这不是一个值得推荐的程序。本程序只是为了帮助读者理解，好的程序应该用数值方法求导数(如例 5－1－1 或例 5－3－4)，或用两分法来求根。读者可自行改进这个程序。

懂得此原理后，实际上不必再去编程，用 fzero 即可求 f(x)的根，要做的工作是把f(x)定义为一个用 M 文件表示的函数，例如建立一个 ex541f.m 文件，其内容为

```
function y=ex541f(x)
y =x.^3+10*x.^2-2*sin(x)-50;
```

再用主程序 x=fzero('ex541f', x0)求解 x。

二、数值定积分

【例 5-4-2】 用数值积分法求 $y=-x^2+115$ 在 x=0 到 x=10 之间所围面积，并讨论步长和积分方法对精度的影响。

解：

◆建模

用欧拉法和梯形法分别求数值积分并作比较。

设将 x 的被积分区间分为 n 段，各段长度为 $\Delta x_i(i=1,\cdots,n)$，包括起点和终点共 n+1 点，算出各点的 y(i)(i=1, …, n+1)，则欧拉法数值积分公式为

$$s=\sum_{i=1}^{n} y(i)\Delta x_i$$

梯形法的公式为

$$q=\sum_{i=1}^{n}\frac{(y(i)+y(i+1))}{2}\Delta x_i$$

◆MATLAB 程序

```
clf,
for dx=[2, 1, 0.5, 0.1]          % 设不同步长
    x=0: .1: 10; y=-x. * x+115; plot(x, y), hold    % 画出被积曲线
    x1=0: dx: 10; n=length(x1);     % 建立自变量数组
    y1=-x1. * x1+115;               % 求取样点上的 y1
    s=sum(y1(1: n-1)) * dx;         % 用欧拉法求积分，注意末尾去掉一点
    q=trapz(y1) * dx;               % 用梯形法求积分
    stairs(x1, y1), plot(x1, y1)    % 画出欧拉法和梯形法的积分区域
    [dx, s, q], pause(1), hold off  % 显示数值
end
```

◆程序运行结果

运行结果如下

步长 dx	欧拉法解 s	梯形法解 q
2	910	810
1	865	815
.5	841.25	816.25
.1	821.65	816.65

用解析法求出的精确解为 2450/3。dx=2 时欧拉法和梯形法的积分面积见图 5-12 所示。在曲线斜率为负的情况下，欧拉法的积分结果一定偏大。梯形法是由各采样点的连线包围的面积，在曲线曲率为负(上凸)时，其积分结果一定偏小，因此精确解在这两者之间。由这结果也能看出，

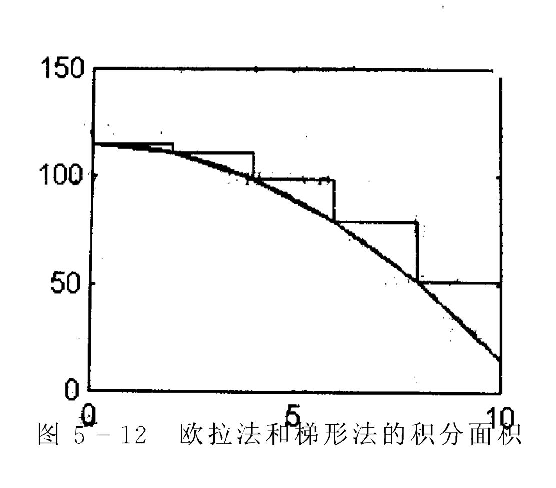


图 5-12　欧拉法和梯形法的积分面积

在步长相同时，梯形法的精度比欧拉法的高。

MATLAB 中还有定积分求面积的函数 quad(见表 4－6)，这个函数的调用语句如下

```
s = quad('ex552f', 0, 10)
```

因此事先必须建立一个函数文件 ex542f.m，其内容为

```
function y = ex542f(x)
y=-x.*x+115;
```

这个文件应存在 MATLAB 的搜索路径上，例如存在 user 子目录中。执行 quad 语句的结果仍为 816.666 7。长格式结果为 816.666 666 666 666 7。

三、多重积分

【例 5－4－3】 计算二重积分

$$\iint_{(\Omega)} (x^2 + y^2)\,dxdy$$

积分区域 Ω 为由 x=1，y=x 及 y=0 所围成的闭合区域。

解：

◆建模

先画出积分区域，如图 5－13 所示，在任意 x 处取出沿 y 向的一个单元条，其宽度为 dx，而高度为 y=x，把它表示为 y 的一个数组，其上的被积函数 f 也是一个数组，沿 y 向的积分可用 trapz 函数完成，得到 s1(k)，它是随 x 而变的。用 for 循环求出所有的 s1(k)，再沿 x 方向用 trapz 函数积分。MATLAB 的数组运算可以代替一个 for 循环，所以二重积分只需用一组 for 语句。

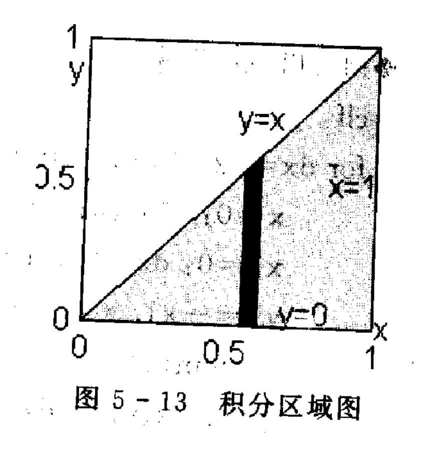


图 5－13　积分区域图

◆MATLAB 程序

```
clear, format compact
fill([0, 1, 1, 0], [0, 0, 1, 0], 'y'), hold          % 画出积分区域
fill([0.55, 0.6, 0.6, 0.55, 0.55], [0, 0, 0.6, 0.55, 0], 'r')    % 画出单元条
dx=input('步长 dx= '); dy=dx;
x=0: dx: 1; lx=length(x);
for k=1: lx
      x1=(k-1)*dx;
      y1=0: dy: x1;
      f=x1.^2+y1.*2;
      s1(k)=trapz(f)*dy;
end
s=trapz(s1)*dx
```

◆程序运行结果

```
s=0.5834
```

改变 dx 的大小，积分的精度也不同。dx 取得愈小，精度就愈高，但运算时间会加长。

【例 5-4-4】 计算三重积分

$$\iiint_{\Omega} xy^2z^3\,dxdydz$$

积分区域 Ω 为由 x=1，y=x，z=xy 及 z=0 所围成的闭合区域。

解：

◆建模

先画出积分区域，如图 5-14 所示，我们参照例 5-2-2 和例 5-2-3 的方法写了一个专门的程序 ex544a 来画这个立体形状。本题中用到的新函数为 line，用以画出与 z 轴平行的平面。另外还有 rotate3d，它是 MATLAB 5 中新创的命令，用它可以使三维图形旋转，以得到最好的视觉效果。

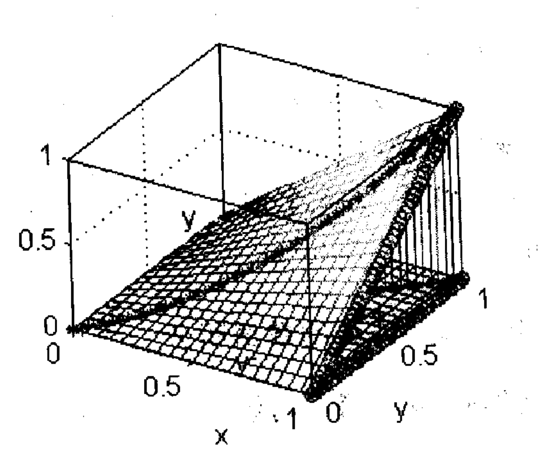


图 5-14　三重积分区域图

在任意[x，y]处取出沿 z 向的一个单元条，其底面积为 dx * dy，而高度为 z=x * y，把它表示为 z 向的一个数组。其上的被积函数 f 也是一个数组，沿 z 向的积分可用 trapz 函数完成，得到 s1。s1 是随 x，y 而变的，先固定 x，用 for 循环求出沿 y 向所有的 s1(j)，用 trapz 函数求其和 s2=trapz(s1)。s2(k)又是随 x 而变的，再沿 x 方向用 trapz 函数积分。由于 MATLAB 的数组运算可以代替一个 for 循环，因此三重积分只用了两组 for 语句，这样使本题的程序比较简明。

◆ MATLAB 程序

1. 绘制积分区域的程序(ex544a)

```
%本程序给出由 x=1，y=x，z=xy 等 3 个曲面以及 z=0 坐标平面围成的积分区域
[x, y]=meshgrid(0: .05: 1);          % 确定计算和绘图的定义域网格
z1=x. * y; z2=zeros(size(z1));       % z=xy 和 z=0 表面方程
mesh(x, y, z1); hold on; mesh(x, y, z2);   % 分别画出两个表面
x1=[0: 0.02: 1]; y1=x1; sx1=length(x1);    % y=x 平面，与 z 平行
zd=[zeros(1, sx1);x1. * y1];      % y=x 平面上，与 z 平行的线族端点 zd=xy
line([x1; x1], [y1; y1], zd];     % 画出此平行线族
plot3([x1; x1], [y1; y1], zd, '*')        % 端点画 *
plot3(ones(2, sx1),[y1;y1],[zeros(1, sx1);y1],'o')   % 画出 x=1 平面上的交
线
xlabel('x'), ylabel('y'), zlabel('z'), hold off,
pause, rotate3d
```

2. 三重积分的 MATLAB 程序(ex544)

```
dx=input('dx='); dy=dx; dz=dx; x=0: dx: 1;
for k=1: length(x)
   x1=(k-1) * dx;
```

```
    y=0: dy: x1;
    for j=1: length(y)
      y1=(j-1) * dy;
      z1=0: dz: x1 * y1;          % z1 数组
      f=x1. * y1.^2. * z1.^3;     % f(z1)
      s1(j)=trapz(f) * dz;        % 沿 z1 积分
    end
    s2(k)=trapz(s1) * dy;         % 沿 y1 积分
  end,
  s=trapz(s2) * dx                % 沿 x1 积分
```

◆程序运行结果

在输入 dx=0.1 时，得 s=0.0030；在输入 dx=0.01 时，得 s=0.0027；精确解为 1/364。

四、微分方程的数值积分

【例 5-4-5】 用数值积分法求解下列微分方程

$$\ddot{y} + y = 1 - \frac{t^2}{2\pi}$$

设初始时间 t0=0；终止时间 tf=3 * pi；初始条件 y(0)=0，$\dot{y}$=0。

解：

◆建模

先将方程化为两个一阶微分方程组，其左端分别为变量 x 的两个元素的一阶导数。

设 $x_1 = y$，$x_2 = \dot{x}_1 = \dot{y}$，则方程可化为

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = -x_1 + 1 - \frac{t^2}{2\pi}$$

写成矩阵形式为

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \left(1 - \frac{t^2}{2\pi}\right) = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \left(1 - \frac{t^2}{2\pi}\right)$$

其中，$\dot{x}$ 为 x 的导数。变量 x 的初始条件为 x(0)=[0；0]，这就是待积分的微分方程组的标准形式。

◆MATLAB 程序

将导数表达式的右端写成一个 ex545f.m 函数程序，内容如下

```
  function xdot=ex545f(t, x)
  u=1-(t·^2)/(pi^2);
  xdot=[0 1; -1 0] * x + [0 1]' * u;
```

主程序调用 MATLAB 中已有的数值积分函数进行积分，其内容如下

```
   clf, t0=0; tf=3 * pi; x0t=[0; 0];          % 给出初始值
   [t, x]=ode23('ex545f', [t0, tf], x0t)      % 此处显示结果
```

```
y=x(:,1);                                    % y 为 x 的第二列
% 本题的解析结果为 y2(I)=(1+2/(pi^2))*(1-cos(t(I)))-t(I)^2/(pi^2)
% 在数值积分输出的时间序列点 t 上计算它的值并画图与数值解作比较
for I=1:length(t);
    y2(I)=(1+2/(pi^2))*(1-cos(t(I)))-t(I)^2/(pi^2);   % 解析解计算
end
u=1-(t.^2)/(pi^2);
clf, plot(t,y,'-',t,u,'+',t,y2,'o')
legend('数值积分解','输入量','解析解') % 图例标注语句
```

◆程序运行结果

程序运行的结果见图 5-15 所示。这个数值积分函数是按精度要求自动选择步长的。它的默认精度为 10^{-3}，因此图中的积分结果和解析解看不出差别。可以用长格式显示 y 和 y2，比较它们的微小误差。若要改变精度要求，可在调用命令中增加可选变元。详情可通过 help ode23 查找。

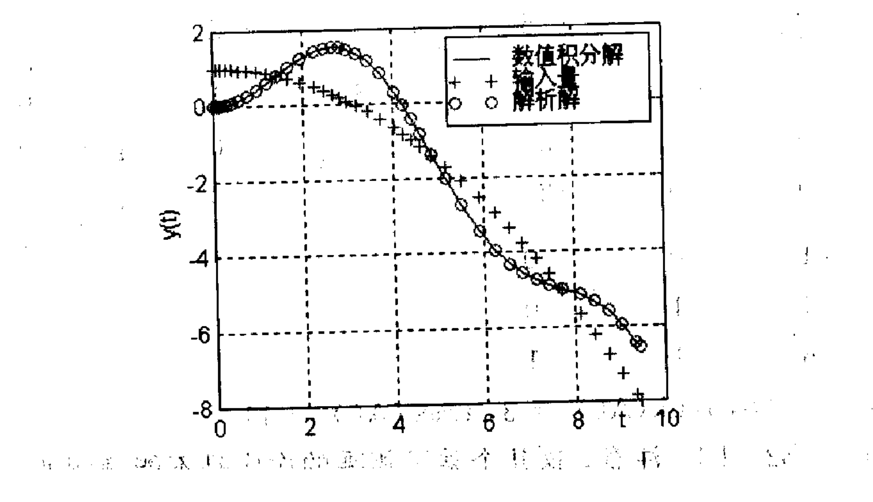


图 5-15　例 5-4-5 中数值积分解与解析解的曲线

5.5　线 性 代 数

【例 5-5-1】　用矩阵的初等变换将 A=[1 0 7; 4 1 5; 2 -1 9] 消元，变为一个上三角阵，求出各相应的初等变换等价的变换乘子 B，并用 MATLAB 检验其行列式的秩和迹。

解：

◆建模

消元法是通过两行元素乘以不同乘子，再相加(或相减)使某个指定的元素数值变为零的方法。消元法等价于一个变换矩阵 B，若 B 把 A 变为 A1，即

B * A=A1

则可用 MATLAB 求得 B=A1/A。

◆MALTAB 程序

```
A=[1 0 7; 4 1 5; 2 -1 9]; A0=A        % 输入 A，并保留一个备份 A0
```

```
A(2, :) = -4 * A(1, :)+A(2, :);
A1=A, B1=A1/A0                  % 消去 A(2, 1), 求 B1
A(3, :) = -2 * A(1, :)+A(3, :);
A2=A, B2=A2/A1                  % 消去 A(3, 1), 求 B2
A(3, :) = -A(3, 2)/A(2, 2) * A(2, :)+A(3, :);
A3=A, B3=A3/A2                  % 消去 A(3, 2), 求 B3
B0 = A3/A0                 % 求三次初等变换的总等价乘子
det(A0), rank(A0), trace(A0),% 用 MATLAB 求原来 A 的行列式、秩和迹
```

◆程序运行结果

```
A1 =      1      0      7        B1=      1      0      0
          0      1    -23                -4      1      0
          2     -1      9                 0      0      1

A2=       1      0      7        B2 =     1      0      0
          0      1    -23                 0      1      0
          0     -1     -5                -2      0      1

A3=       1      0      7        B3 =     1      0      0
          0      1    -23                 0      1      0
          0      0    -28                 0      1      1

B0 =      1      0      0
         -4      1      0
         -6      1      1

det(A0) = -28, rank(A0) = 3, trace(A0) = 11
```

显然 B0=B3 * B2 * B1。注意，这几个乘子相乘的次序是不能颠倒的。

很容易用观察法判断结果的正确性，因为对上三角阵而言，行列式是其主对角线元素的乘积，迹是其主对角线元素的和，秩是其主对角线上非零元素的数目。

读者可进一步做以下工作：

(1) 改变前几行程序，使本程序能用于任何用户输入的 3×3 矩阵。

(2) 重写一个能把任何 4×4 矩阵化为上三角矩阵的程序。

(3) 再用列向的消元法，将 A3 化为对角矩阵，并确定其等价(右)乘子。

(4) 求出把此对角矩阵化为单位矩阵所需的乘子，讨论 A 的逆阵的求法。

【例 5-5-2】 hilbert 矩阵 H 具有如下规律，以三阶 hilbert 矩阵 H3 为例。

```
H3 =    1      1/2     1/3
       1/2     1/3     1/4
       1/3     1/4     1/5
```

它可由 MATLAB 中的语句 format rat，H3=hilb(3)产生。hilbert 矩阵的一个特点是它的行列式很小，条件数很大，很接近于奇异阵，阶次愈高愈接近奇异，试用它组成线性方程组，探讨其误差与奇异性的关系。

解：

◆建模

要构成一个 n 元一阶线性方程组 Hn * x＝bn，其中 x 的 n 个分量均为 1，需做如下工作：

(1) 求 Hn 的行列式及条件数。

(2) 求 bn。

(3) 按方程 Hn * x1＝bn，求 x1，并与 x 相比较，求出其最大相对误差(注意用长格式显示)。

解：

◆MATLAB 程序(n＝4 时)

```
H4 = hilb(4), r = rank(H4), c = cond(H4)
x=ones(4, 1); b4 = H4 * x ,
format long, x1 = H4   b4,
e = max(abs(x1-x))
```

◆程序运行结果

```
H4 =     1.0000     0.5000     0.3333     0.2500
         0.5000     0.3333     0.2500     0.2000
         0.3333     0.2500     0.2000     0.1667
         0.2500     0.2000     0.1667     0.1429
r = 4
c = 1.5514e+004
b4 =     2.0833
         1.2833
         0.9500
         0.7595
x1 =   1.00000000000003
       0.99999999999970
       1.00000000000072
       0.99999999999953
e =    7.1987e-013
```

◆讨论

H4 的条件数达 10 000 以上，而解的相对误差在 10^{-12} 以下，可见 MATLAB 的计算精度很高。读者可自行扩展此程序，把 H 矩阵的阶次逐步提高，看到什么阶次相对误差会达到 100%，到什么阶次 MATLAB 会发出警告。

[例 **5－5－3**]　设有对称实矩阵

$$a = [\ 2, 4, 9; 4, 2, 4; 9, 4, 18\ ]$$

试求其特征根和特征向量，并讨论其特性。

解：

◆MATLAB 程序

```
a = [ 2, 4, 9; 4, 2, 4; 9, 4, 18 ],
```

```
[v, d] = eig(a)      % v是特征向量，d是以特征根为元素的对角阵
det(v), det(d), trace(a), trace(d),
```

◆运行结果

```
v =      0.8483     -0.3290      0.4149
        -0.4514     -0.8589      0.2419
        -0.2767      0.3925      0.8771
d =     -3.0645           0           0
              0      1.7042           0
              0           0     23.3603
```

det(a) = −122, det(d) = −122.0000, trace(a) = 22, trace(d) = 22,

◆讨论

特征根的和仍为原矩阵的迹，特征根的积仍为原矩阵的行列式。不难检验其特征根和特征向量的如下关系：

```
(a-d(1, 1) * eye(3)) * v(:, 1)=0
(a-d(2, 2) * eye(3)) * v(:, 2)=0
(a-d(3, 3) * eye(3)) * v(:, 3)=0
```

也可写成更简练的包含 3 个特征根的矩阵关系

```
a * v-v * d=0
```

第 6 章 在普通物理中的应用举例

6.1 物理数据处理

【例 6-1-1】 写出一个程序，能把用户输入的摄氏温度转为华氏，反之也可。

解：

◆建模

两种温度单位之间的转换公式为：

摄氏变华氏 $Tout=\frac{9}{5}Tin+32$

华氏变摄氏 $Tout=(Tin-32)\frac{5}{9}$

在程序中要先考虑由用户选择转换的方向，以确定选用的公式。

◆MATLAB 程序

```
k = input('选择 1：摄氏变华氏；选择 2：华氏变摄氏；键入数字：')；
Tin = input('输入待变换的温度(允许输入数组)：')；
if    k==1 Tout = Tin * 9/5 +32，           % 摄氏转华氏
elseif k==2 Tout = (Tin-32) * 5/9，         % 华氏转摄氏
else disp('未给转换方向，转换无效')，end
s = ['华氏'，'摄氏']；
s1=['转换后的温度为',s(k，:),num2str(Tout),'度'] %注意这个语句的编写方法
```

◆结论

这是一个简单的例子，只涉及两种单位之间的转换，如果涉及很多单位之间的相互变换，就要多一些设计技巧，下面的例子将提供一种方法。

【例 6-1-2】 写出一个程序，能把用户输入的长度单位在厘米、米、千米、英寸、英尺、英里、市尺、市里之间任意转换。

解：

◆建模

这里采取的技巧分成两步，第一步把输入量变换为米，第二步再把米变换为输出单位。另外，把变换常数直接表示为一个数组，选择单位的序号也就成了数组的下标，这样，程序就比较简明易读。

◆MATLAB 程序(长度单位换算程序 ex612.m)

```
clear all;      disp(' 长度单位换算程序')
fprintf('长度单位: n');              % 选择输入输出的单位
fprintf(' 1) 厘米   2) 米   3) 千米   4) 英寸   n');   %   n 是换行符
fprintf(' 5) 英尺   6) 英里   7) 市尺   8) 市里   n');
InUnits = input('选择输入单位: ');
OutUnits = input('选择输出单位: ');
% 设定各种单位对米的变换常数数组 ToMeter
ToMeter = [0.01, 1.00, 1000.0, 0.0254, 0.3048, 1609.3, 1/3, 500 ];
  FrmMeter= 1./ ToMeter; % 反变换常数数组 FrmMeter 为 ToMeter 数组的倒数
Value = input('输入待变换的值(0 为退出): ');
while( Value ~= 0 )
  ValueinM = Value * ToMeter(InUnits);              % 把输入值变为米
  NewValue = ValueinM * FrmMeter(OutUnits);         % 把米变为输出单位
  fprintf('变换后的值是 %g   n', NewValue);          % 打印变换后的值
  Value = input('输入待变换的值(0 为退出): ');        % 提问下个输入值
end
```

◆程序运行结果

程序在运行后将向用户提问三次，若选输入单位为米，输出单位为英寸，待变换的值为 2 米，则变换后的值是 78.7401 英寸。

【例 6-1-3】 设给某元件加[1, 2, 3, 4, 5]V 电压，测得的电流为[0.2339, 0.3812, 0.5759, 0.8153, 0.9742]mA，求此元件的电阻。

解:

◆建模

设直线的方程为 y= a(1)x +a(2)，待定的系数是 a(1)，a(2)。将上述数据分别代入 x、y，得:

$$a(1)+a(2)=0.2339$$
$$2a(1)+a(2)=0.3812$$
$$3a(1)+a(2)=0.5759$$
$$4a(1)+a(2)=0.8153$$
$$5a(1)+a(2)=0.9742$$

用矩阵形式表达这五个联立方程，其系数矩阵设为 datax 和 datay，有

```
datax * a(1) + ones(N, 1) * a(2) = datay
```

其中 datax、datay 都是 N 行数据列向量，这是 N 个一次代数方程，含两个未知数，方程数超过未知数个数，是一个超定方程，写成 A * a = B，其最小二乘解可以直接使用 MATLAB 的左除运算符 a = A B 来求得。因此程序为

```
A = [datax, ones(N, 1)]; B = datay; a = A  B
```

a(2)的存在说明此直线不通过原点。若要在过原点的曲线族中拟合，就要在原始方程

中规定 a(2)＝0。把 A 中的第二列去掉，即令 A ＝ datax，a0 ＝ A　B，如果需要用二次曲线(抛物线)来拟合数据，则结果为

A ＝ [datax.^2，datax，ones(N，1)]；B ＝ datay；a ＝ A　B

用 4.3 节中的 polyfit 函数就更加简单，无需列写 A。如果用 n 次曲线拟合，公式为 an ＝ polyfit(datax,datay,n)，在知道系数多项式 an 后，求多项式值的语句为 yi＝polyval(an，xi)。

◆MATLAB 程序

```
clear
datax = [1: 5]';
datay=[ 0.2339, 0.3812, 0.5759, 0.8153, 0.9742 ]';          % 原始数据
A = [datax, ones(5, 1)]; B = datay; a = A  B, r=1/a(1) % 线性拟合
plot(datax, datay, 'o'), hold on              % 绘出原始数据点图
xi=0: 0.1: 5; yi=a(1) * xi+a(2);              % 设置 51 个取值点
A1 = datax, a0 = A1     B,                    % 通过原点的线性拟合
plot(xi, yi, xi, a0 * xi, ':')                % 绘图
a2 = polyfit(datax,datay,2); yi=polyval(a2,xi);        % 二次拟合
plot(xi, yi)                    % 绘二次拟合曲线
hold off
```

◆程序运行结果

```
a = 0.1905
    0.0247
a0 = 0.1972
a2 = 0.0041
     0.1657
     0.0536
```

绘出的三种拟合曲线如图 6－1 所示。

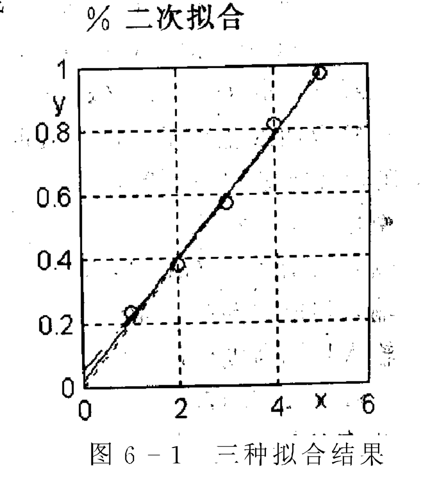


图 6－1　三种拟合结果

6.2　力　学　基　础

【例 6－2－1】　设目标相对于射点的高度为 y_f，给定初速，试计算物体在真空中飞行的时间和距离。

解：

◆建模

无阻力抛射体的飞行是中学物理就解决了的问题，本题的不同点是目标和射点不在同一高度上，用 MATLAB 可使整个计算和绘图过程自动化。其好处是可快速地计算物体在不同初速和射角下的飞行时间和距离。关键在求落点时间 t_f 时，需要解一个二次线性代数方程。

由

$$y=v_0\sin\theta_0\cdot t-\frac{1}{2}gt^2=y_f$$

解出 t，它就是落点时间 t_f。t_f 会有两个解，我们只取其中一个有效解。再求

$$x_{max}=v_0\cos\theta_0\cdot t_f$$

◆MATLAB 程序

```
clear; y0 = 0; x0 = 0;                          % 初始位置
vMag = input('输入初始速度 (m/s): ');            % 输入初始速度的大小和方向
vDir = input(' 输入初速方向(度): ');
yf = input('输入目标高度(m): ');                  % 输入目标高度 yh
vx0 = vMag * cos(vDir * (pi/180));              % 计算 x, y 方向的初始速度
vy0 = vMag * sin(vDir * (pi/180));
wy = -9.81; wx = 0;                             % 重力加速度 (m/s^2)
tf=roots([wy/2, vy0, y0-yf]);           % 解二次线性代数方程,计算落点时间 tf
tf=max(tf);                                     % 去除 tf 两个解中的庸解
t=0: 0.1: tf;
y = y0 + vy0 * t + wy * t.^2/2;                 % 计算轨迹
x = x0 + vx0 * t + wx * t.^2/2;
xf = max(x), plot(x, y),                        % 计算射程，画出轨迹
```

在检查曲线正确后，键入 hold 命令，把曲线保留下来，以使用同样的初速，不同的射角，比较其轨迹和最大射程。

◆程序运行结果

输入初始速度 (m/s): 50

输入初速方向(°): 40

输入目标高度(m): 8

  xf = 237.4738

初速方向为 50°时

  xf = 241.0454

所得曲线如图 6-2 所示。可以随后键入下列命令来设置坐标网格和进行标注。

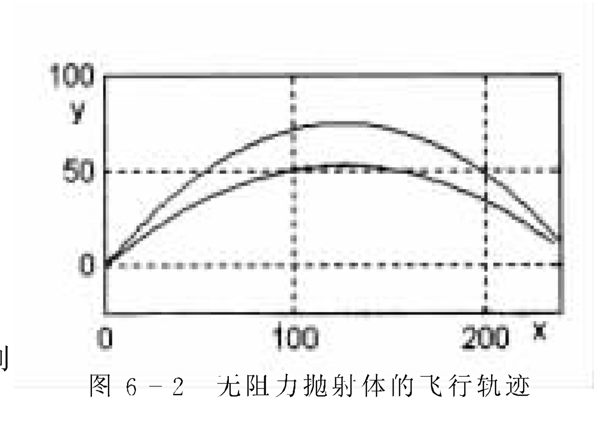


图 6-2 无阻力抛射体的飞行轨迹

```
grid, gtext('高低角 50'), gtext('40')
```

【例 6-2-2】 给定质点沿 x 和 y 两方向的运动规律 x(t) 和 y(t)，求其运动轨迹，并计算其对原点的角动量。

解：

◆建模

本例要求用户输入运动规律的解析表示式，这需要用到字符串的输入语句，应当在 input 语句中加上第二变元's'，而运行这个字符串要用 eval 命令。当 x(t) 和 y(t)都是周期运动时，所得的曲线就是李萨如图形。角动量等于动量与向径的叉乘(cross product)。求速度需要用导数，可用 MATLAB 的 diff 函数作近似导数计算。设角动量为 $\vec{L}$，质点的动量为 $\vec{P}=m\vec{v}$，向径为 $\vec{r}$，则

$$\vec{L}=\vec{r}\times\vec{P}=\vec{r}\times m\vec{v}$$

在 XY 平面上有

$$L = x \cdot mv_y - y \cdot mv_x$$

◆MATLAB 程序

```
clear all;
% 读入字符串，它应是满足无素群运算的语句
x = input('x=', 's'); y = input('y=', 's');
tf = input(' tf= '); m=1          % 设定质量 m，此处设为 1
Ns=100; t=linspace(0, tf, Ns); dt=tf/(Ns-1);% 分 Ns 个点，求出时间增量 dt
xPlot=eval(x); yPlot=eval(y); % 计算 Ns 个点的位置 x(t), y(t)
% 计算各点 x(t), y(t)的近似导数和角动量，注意导数序列长度比原函数少 1
   px = m * diff(xPlot)/dt;        % px = M dx/dt
   py = m * diff(yPlot)/dt;        % py = M dy/dt
   LPlot = xPlot(1: Ns-1). * py - yPlot(1: Ns-1). * px;    %求角动量
   % 画出轨迹及角动量随时间变化的曲线
clf; figure(gcf);                  % 清图形窗并把它前移
subplot(1, 2, 1), plot(xPlot, yPlot);          % 画点的轨迹图
axis('equal'); grid               % 使两轴比例相同
subplot(1, 2, 2), plot(t(1: Ns-1), LPlot);       % 画动量矩随时间的曲线
```

◆程序运行结果

运行此程序，输入

```
x=t. * cos(t)
y=t. * sin(t)
tf=20
```

后，得出图 6 - 3，读者可看出其角动量单调递增。如果输入

```
x=cos(2 * t)
y=sin(3 * t)
tf=5
```

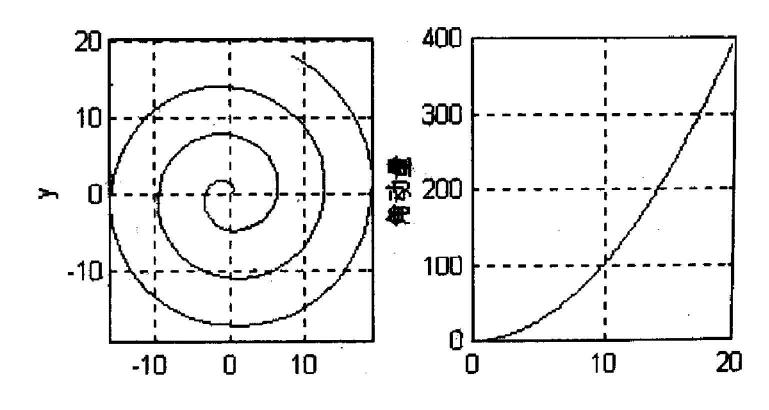


图 6 - 3 按方程 x=t cos(t)，y=t sin(t)画出轨迹及角动量曲线

则得到图 6 - 4，其轨迹图就是李萨如图形。

读者可输入其它形式的 x(t)和 y(t)，探讨其结果。注意输入式一定要满足对 t 作元素

群运算的格式。

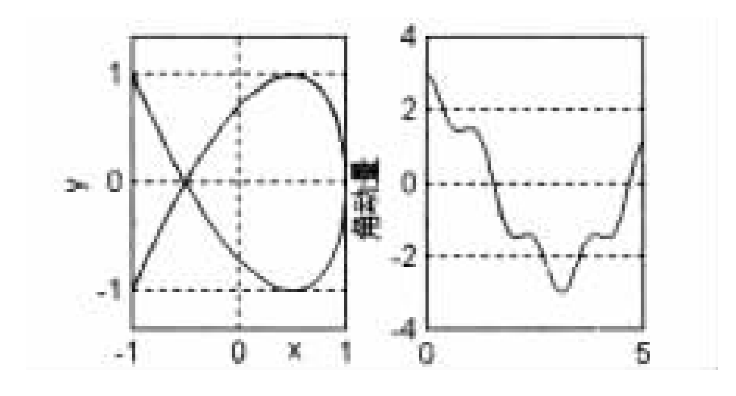


图 6-4 按方程 $x=\cos(2t)$，$y=\cos(3t)$画出轨迹及角动量曲线

【例 6-2-3】 物体 A(质量为 m_1)在具有斜面的物体 B(质量为 m_2)上靠重力下滑，设斜面和地面均无摩擦力，求 A 沿斜面下滑的相对加速度 a_1 和 B 的加速度 a_2，并求斜面和地面的支撑力 N_1 及 N_2。

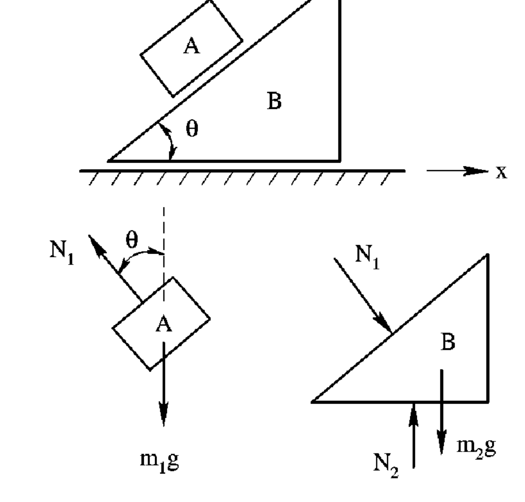


图 6-5 物体受力图

解：

◆建模

分别画出 A 和 B 的受力图如图 6-5 所示。

对物体 A 列写动力学方程，注意它的绝对加速度是 a_2 与 a_1 的合成

$$m_1(a_1\cos\theta-a_2)=N_1\sin\theta \qquad (1)$$

$$m_1a_1\sin\theta=m_1g-\cos\theta\cdot N_1 \qquad (2)$$

对物体 B

$$m_2a_2=N_1\sin\theta \qquad (3)$$

$$N_2-N_1\cos\theta-m_2g=0 \qquad (4)$$

四个方程包含 a_1，a_2，N_1，N_2 等四个未知数，将含未知数的项移到左边，常数项移到等式右边，得到

$$\begin{bmatrix} m_1\cos\theta & -m_1 & -\sin\theta & 0 \\ m_1\sin\theta & 0 & \cos\theta & 0 \\ 0 & m_2 & -\sin\theta & 0 \\ 0 & 0 & -\cos\theta & 1 \end{bmatrix}\cdot\begin{bmatrix} a_1 \\ a_2 \\ N_1 \\ N_2 \end{bmatrix}=\begin{bmatrix} 0 \\ m_1g \\ 0 \\ m_2g \end{bmatrix}$$

写成 $A\cdot X=B$

故有 $X=A\quad B$

◆MATLAB 程序

```
m1=input('m1= '); m2=input('m2= ');
theta=input('theta(度)= ');
theta=theta*pi/180; g=9.81;
A = [m1*cos(theta), -m1, -sin(theta), 0; ...
        m1*sin(theta), 0, cos(theta), 0; ...
              0       , m2, -sin(theta), 0; ...
```

```
                0        , 0, -cos(theta), 1 ];
B = [ 0, m1 * g, 0, m2 * g]'; X=A\B;
a1=X(1), a2=X(2), N1=X(3), N2=X(4)
```

◆程序运行结果

输入 m1=2，m2=4，及 theta=30，得到

```
a1 =6.5400        a2 =1.8879
N1 = 15.1035      N2 = 52.3200
```

静力学平衡和动力学中求力与加速度关系的问题，通常都可归结为线性方程组的求解，只要方程组列写正确，用 MATLAB 的矩阵除法就可以方便而准确地求解。

【例 6-2-4】 质量为 m 的小球以速度 u_0 正面撞击质量为 M 的静止小球，假设碰撞是完全弹性的，即没有能量损失，求碰撞后两球的速度，及它们与两球质量比 K=M/m 的关系。

解：

◆建模

设碰撞后两球速度都与 u_0 同向，球 m 的速度为 u，球 M 的速度为 v，列出动量守恒和能量守恒方程，引入质量比 K=M/m 和无量纲速度 $u_r=u/u_0$，$v_r=v/u_0$ 后，有

动量守恒 $$mu_0=mu+Mv \tag{1}$$

动能守恒 $$\frac{1}{2}mu_0^2=\frac{1}{2}mu^2+\frac{1}{2}Mv^2 \tag{2}$$

化为 $$Kv_r+u_r=1 \tag{3}$$

$$Kv_r^2+u_r^2=1 \tag{4}$$

由(3) $$v_r=\frac{1-u_r}{K} \tag{5}$$

代入(4) $$(1-u_r)^2+Ku_r^2=K \tag{6}$$

主动球的能量损失为

$$E_m=\frac{1}{2}m(u_0^2-u^2)=\frac{m}{2}u_0^2(1-u_r^2)$$

展开并整理多项式(6)，得

$$(1+\frac{1}{K})u_r^2-\frac{2}{K}u_r+\left(\frac{1}{K}-1\right)=0$$

可用 roots 命令求根 u_r。

◆MATLAB 程序(ex624.m)

```
clear
K=logspace(-1, 1, 11);   % 设自变量数组 K，从 K=0.1~10，按等比取 11 个
点
for i=1:length(K)            % 对各个 K 循环计算
ur1=roots([(1+1/K(i)), -2/K(i), (1/K(i)-1)]);   % 二次方程有两个解
ur(i)=ur1(abs(ur1-1)>0.001);                  % 去掉在 1 邻近的庸解
end
vr=(1-ur)./K;                                 % 用(5)式求 vr，用元素群运算
```

```
em=1-ur.*ur                        % 主动球损失的能量(相对值)
[ur; vr]                           % 显示输出数据
semilogx(K,[ur,vr,em]),grid        % 绘图
```

◆程序运行结果

数字结果为(省略了几行)

K	ur	vr	em
0.1000	0.8182	1.8182	0.3306
0.3981	0.4305	1.4305	0.8147
1.0000	0	1.0000	1.0000
2.5119	−0.4305	0.5695	0.8147
10.000	−0.8182	0.1818	0.3306

绘出的曲线如图 6-6 所示。

可以看出,当 K>1 时,u_r 为负,即当静止球质量大于主动球质量时,主动球将产生回弹。当 K=1 时,$u_r=0$,即主动球将全部动能传给静止球。当K<1 时,u_r 为正,说明主动球将继续沿原来方向运动。

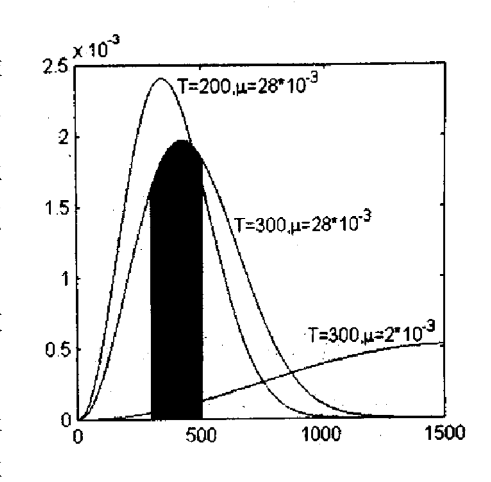


图 6-6　弹性碰撞后球速与 K 的关系

在宏观世界中很难找到完全弹性碰撞,而在微观世界中,上述结果可以用来解释康普顿效应,即光子撞击电子后,其散射光波长会变长,而且波长的增加量与散射角有关。因为光子的动能为 hυ,其中 h=6.63e−34 为普朗克常数,而 υ 是光子的角频率,如上所述,在弹性碰撞后,光子损失了一些能量,必然表现为 υ 减小,即波长增加。本例只考虑了正面碰撞,分析了不同质量比的影响。要解释康普顿效应,必须考虑光子的散射是由侧面碰撞产生的,这时可以把质量比取定(光子的静止质量为零,应考虑它的动质量 $h\upsilon/c^2$,c 为光速),可以分析出其损失的能量(因而其波长)与散射方向相关,读者可自行扩展此程序进行研究。

6.3　分子物理学和热力学

【例 6-3-1】 利用气体分子运动的麦克斯韦速度分布律,求 27℃下氮分子运动的速度分布曲线,并求速度在 300~500 m/s 范围内的分子所占的比例,讨论温度 T 及分子量 μ 对速度分布曲线的影响。

解:

◆建模

麦克斯韦速度分布律为

$$f=4\pi\left(\frac{m}{2\pi kT}\right)^{3/2}\cdot v^2\cdot \exp\left(\frac{-mv^2}{2kT}\right)$$

本例将说明如何从复杂数学公式绘制曲线，并研究单个参数的影响。先把麦克斯韦速度分布律列成一个子程序，以便经常调用，并把一些常用的常数也放在其中，这样主程序就简单了。

◆MATLAB 程序

1. 子程序(mxw1.m)

```
function f=mxwl(T, mu, v)
% 其中：
% mu——分子量，公斤·摩尔⁻¹(如氮为 28×10⁻³)，即 μ
% v——分子速度(可以是一个数组)
% T——气体的绝对温度
R=8.31 ;                  % 气体常数
k=1.381*10^(-23);         % 玻尔茨曼常数
NA=6.022*10^23;           % 阿伏伽德罗数
m=mu/NA;                  % 分子质量
% 麦克斯韦分布率
f=4*pi*(m/(2*pi*k*T)).^(3/2).*exp(-m*v.^2./(2*k*T)).*v.*v;
```

2. 主程序(ex631.m)

```
T=300; mu=28e-3;      % 给出 T, mu
v=0:1500;             % 给出自变量数组
y=mxwl(T, mu, v);     % 调用子程序
plot(v, y), hold on   % 画出分布曲线
v1=300:500;           % 给定速度范围
y1=mxwl(T, mu, v1);   % 该范围的分布
% 画出该曲线所围区域
fill([v1, 500, 300], [y1, 0, 0], 'r')
trapz(y1)   % 求该范围概率积分
```

◆程序运行结果

执行此程序所得曲线及填充图，如图 6-7 所示。

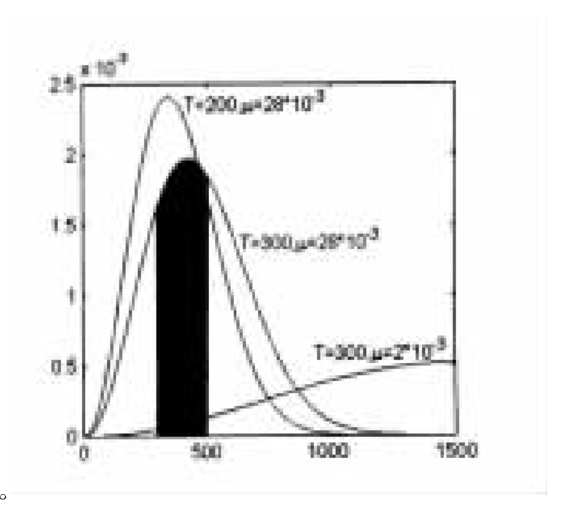


图 6-7　麦克斯韦分布曲线

积分结果为

```
ans = 0.3763
```

为了看出 T 和 mu 对曲线形状的影响，可键入 hold，再在程序中加上语句：

```
T=200; mu=28e-3; y=mxwl(T, mu, v); plot(v, y)   % 改变 T，画曲线
T=300; mu=2e-3; y=mxwl(T, mu, v); plot(v, y)    % 改变 mu，画曲线
```

按鼠标指定的位置加上标注字符

从图 6-7 中可见，减小 T，使分子的速度分布向低端移动；减小分子量 mu，使速度分布向高端移动，这是与物理概念相一致的。

【例 6-3-2】 编写反映热力学过程(包括等压、等容和等温三种)P－V 图的程序，并计算各过程中所做的功。

解：

◆建模

按照理想气体方程

$$PV=nMoles \cdot RT$$

其中，nMoles是研究对象（气体）的摩尔数，即表示气体的数量，因为1 mol的气体在标准状态[$T=273K$，$P=1.013\times10^5$(Pa)]的容积是0.0224 m^3，$R=8.31$是气体常数。因此上式中有P、V、T等三个变量，必须再加一个约束条件，才能使气体的热力过程有确定的规律，通常用P－V图上的轨迹来表示这种规律。该轨迹下的面积$Q=\int P\,dV$为所做的功。

本题中的三个约束条件是：

(1) 等压路径，P＝常数，在P－V图上是一根水平线，它所做的功为$P(V_2-V_1)$。

(2) 等容路径，V＝常数，在P－V图上是一根垂直线，因此所做的功为零。

(3) 等温路径，T＝常数，在P－V图上是一根双曲线，(P·V＝常数)它所做的功为

$$W=Q=nMoles \cdot RT \cdot \ln\left(\frac{V_2}{V_1}\right)$$

实际上还有一条很重要的路径，即绝热路径。读者可自行尝试补充到程序中去，此时还得补充一个热容比γ，作为气体的参数输入。

◆MATLAB程序

```
% 变量和图形初始化，输入气体的摩尔数 nMoles，初始压力 P(1)，初始容积 V(1)，
% 气体常数 R=8.314；给定起始总功 WTotal=0；点序号 iPoint = 1；画等温曲线用
% 的点数 NCurve = 100；P－V 图第一点坐标 PPlot = P(1)；VPlot = V(1)
T(1) = P(1) * V(1)/(nMoles * R);        % 算出初始温度
% 为了进入循环，先要设两个不相等的 PathType 和 QuitType 值
QuitType = 4; PathType = 0;
while(PathType ~= QuitType)    % 在菜单上选择'退出'之前不断循环
  % 选择路径类型或退出
  iPoint = iPoint + 1;                        % 下一点
  fprintf('对过程 #%g    n', iPoint-1);
  PathType = menu(sprintf('过程 %g：选择下一路径', iPoint-1), ...
      '等压', '等容', '等温', '退出');         % 图形界面菜单生成语句
  switch PathType
  case 1                  % 等压路径
    V(iPoint) = input('输入新容积：');
    P(iPoint) = P(iPoint-1);        % 压力不变
    T(iPoint) = P(iPoint) * V(iPoint)/(nMoles * R); % 按新容积算出温度
    W = P(iPoint) * (V(iPoint) - V(iPoint-1));    % 计算等压过程所做的功
    VPlot = [VPlot, V(iPoint)];    % 加上新的容积和压力点，用以绘图
    PPlot = [PPlot, P(iPoint)];
  case 2                  % 等容路径
```

```
        P(iPoint) = input('输入新压力：');
        V(iPoint) = V(iPoint-1);   % 容积不变
        T(iPoint) = P(iPoint) * V(iPoint)/(nMoles * R); %按新压力算出温度
        W = 0;                     % 等容路径上所做的功为零
        VPlot = [VPlot, V(iPoint)]; % 加上绘图用的新容积和压力点
        PPlot = [PPlot, P(iPoint)];
      case 3                       % 等温路径
        V(iPoint) = input('输入新容积：');
        T(iPoint) = T(iPoint-1);             % 温度不变
        P(iPoint) = nMoles * R * T(iPoint)/V(iPoint); % 按新容积求新压力
        W = nMoles * R * T(iPoint) * log(V(iPoint)/V(iPoint-1));   % 求所做的
功
        % 用元素群运算求等温路径上的 P 和 V，加进绘图数据中
        VNew = linspace(V(iPoint-1), V(iPoint), NCurve);
        PNew = nMoles * R * T(iPoint)./VNew;
        VPlot = [VPlot, VNew];          % 将新的 V，P 点加入绘图数据中
        PPlot = [PPlot, PNew];
      otherwise
      end
      % 画出到目前为止的 P-V 图
      if( PathType ~= QuitType )
        WTotal = WTotal + W;                  % 将新做的功加进总功
        figure(gcf); plot(V, P, 'o', VPlot, PPlot, '-') % 图形窗移前，绘图
        标注语句略
      end
    end
```

◆程序运行结果

程序运行时，屏幕上先出现一个空的图形窗，用鼠标拖动图框边缘，将它缩至屏幕右上角，其大小如图 6-8 所示。

按程序提问输入初始值：

nMoles=0.5

P=le5 Pa

V=0.012 m^3

这时，屏幕上出现图 6-9 所示控制界面。

在图 6-9 中，依次输入：

第一段为“等温”，V(2)=0.002；

第二段为“等容”，P(3)=1.5e6；

第三段为“等温”，V(4)=0.012；

第四段为“等容”，P(5)=1.0e5。

得出图 6-8 所示的热力循环曲线。

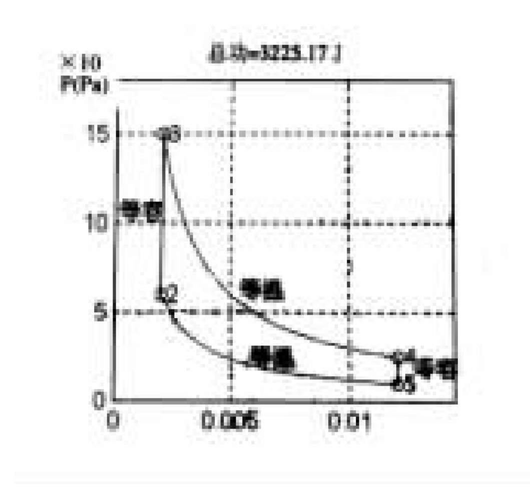


图 6-8　热力循环曲线及其所做的功

该循环所做的总功为

WTotal = 3.2252e+003(J)

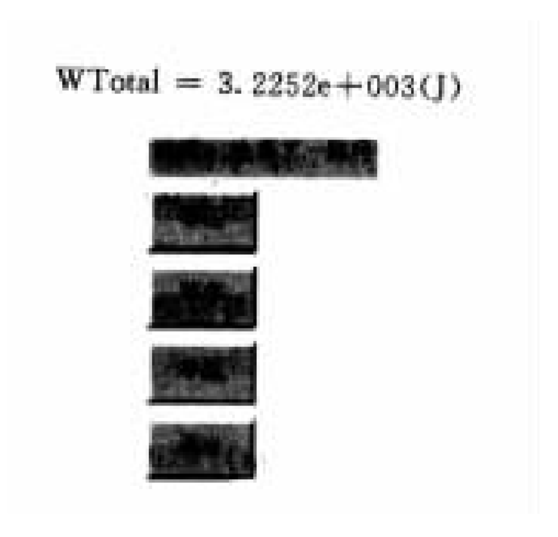


图 6-9 图形控制界面

6.4 静 电 场

【例 6-4-1】 计算平面上 N 个电荷之间的库仑引力。

解：

◆建模

按库仑定律，q_2 对 q_1 的引力公式为 $F = q_1q_2/4\pi r^2\varepsilon_0$，方向与 q_2 到 q_1 的矢径 r 相同，因此 q_1q_2 同号时，该力是斥力。

其分量的公式可写成：

$$F_x = \frac{q_1q_2(x_2-x_1)}{4\pi r^3\varepsilon_0}$$

$$F_y = \frac{q_1q_2(y_2-y_1)}{4\pi r^3\varepsilon_0}$$

$$r = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

编写程序时，先输入电荷的数目、各电荷的坐标及电荷量，再选定一个电荷，求其它电荷对它的作用力，叠加求合力。再选下一个电荷，依此类推。

◆MATLAB 程序

```
clear all;
N = input('输入电荷数目 N=：');
for ic=1:N                        % 输入给定条件
  fprintf('------    n 对电荷 #%g    n', ic);
  rc = input('输入电荷位置[x y](米)：');
  x(ic) = rc(1);                  % 电荷 ic 的 x 坐标
  y(ic) = rc(2);                  % 电荷 ic 的 y 坐标
  q(ic) = input('输入电荷量(库仑)：');
```

```
end
E0 = 8.85e-12;              % 真空中的常数 ε0 (C^2/(N m^2))
C0 = 1/(4 * pi * E0);       % 合并常数
for ic = 1:N                % 循环计算每个电荷所受的力
  Fx = 0.0; Fy = 0.0;       % 先把力初始化为零
  for jc = 1:N              % 求其它电荷给第 ic 个电荷的力
    if( ic ~= jc )          % 若电荷 jc 不是 ic 自身
      % 计算两电荷间的距离
      xij = x(ic) - x(jc); yij = y(ic) - y(jc);
      Rij = sqrt(xij^2 + yij^2);        % 斜距
      % 用库仑定律计算此两电荷的引力的分量叠加到 Fx, Fy 中
      Fx = Fx + C0 * q(ic) * q(jc) * xij/Rij^3;
      Fy = Fy + C0 * q(ic) * q(jc) * yij/Rij^3;
    end
  end
  fprintf('其它电荷作用在电荷 #%g 上的合力为:  n', ic);   %显示结果
  fprintf(' x-分量: %g N   n', Fx);
  fprintf(' y-分量: %g N   n', Fy);
end
```

在本程序中要注意学会循环提示并输入参数的方法，以及用双循环解决较复杂的计算过程的编程方法。

【例 6-4-2】 设电荷均匀分布在从 $z=-L$ 到 $z=L$，通过原点的线段上，其密度为 q (单位为 C/m)，试求出在 xy 平面上的电位分布。

解：

◆建模

点电荷产生的电位可表示为 $V = Q/4\pi r\varepsilon_0$，它是一个标量。其中 r 为电荷到测量点的距离。线电荷所产生的电位可用积分或叠加的方法来求。为此把线电荷分为 N 段，每段长为 dL(在 MATLAB 中，由于程序只认英文字母，dL 应理解为 ΔL)。每段上电荷为 q * dL，看做集中在中点的点电荷，它产生的电位为

$$dV=\frac{q\mathrm{d}L}{4\pi r\varepsilon_0}$$

然后对全部电荷求和即可。

把 xy 平面分成网格，因为 xy 平面上的电位仅取决于离原点的垂直距离 R，所以可以省略一维，只取 R 为自变量。把 R 从 0～10 m 分成 Nr+1 点，对每一点计算其电位。

◆MATLAB 程序

```
输入线电荷半长度 L，分段数 N，Nr 及线电荷密度 q 的语句
E0=8.85e-12;              % 真空电介质常数 ε0
C0=1/4/pi/E0 ;            % 归并常数
L0=linspace(-L, L, N+1);       % 将线电荷分 N 段
```

```
    L1=L0(1:N);L2=L0(2:N+1);   % 确定每个线段的起点和终点
    Lm=(L1+L2)/2;dL= 2 * L/N;  % 确定每个线段的中点坐标,Lm 是 N 元数组
    R=linspace(0,10,Nr+1);     % 将 R 分 Nr+1 点
    for k=1:Nr+1               % 对 R 的 Nr+1 点循环计算
        Rk=sqrt(Lm.^2+R(k)^2); % 测量点到各电荷段的向径长度,N 元数组
        Vk=C0 * dL * q./Rk;    % 各电荷段在测量点处产生的电位,N 元
        V(k)=sum(Vk);          % 对各电荷段在测量点处产生的电位数组求和
    end
    [max(V),min(V)]            % 显示最大最小电位
    plot(R,V),grid             % 绘图
```

◆程序运行结果

运行此程序,输入:

(1) q=1,L=5,N=50,Nr=50

(2) q=1,L=50,N=500,Nr=50

所得电场的最大值和最小值分别为:

(1) 1.0e+010 * [9.3199,0.8654]

(2) 1.0e+011 * [1.3461,0.4159]

沿 R 的电场分布如图 6-10 所示。

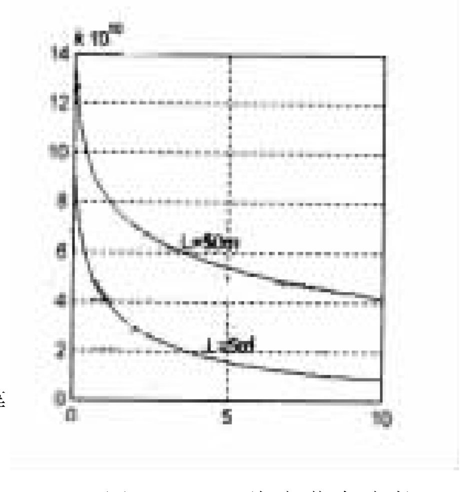


图 6-10 线电荷产生的静电位分布

【例 6-4-3】 由电位的表示式计算电场并画出等电位线和电场方向。

解:

◆ 建模

如果已知空间的电位分布

$$V=V(x,y,z)$$

则空间的电场等于电位场的负梯度

$$\vec{E}=-\text{gradient}(v)=\left(\frac{dv}{dx}\vec{i}+\frac{dv}{dy}\vec{j}+\frac{dv}{dv}\vec{k}\right)$$

其中,$\vec{i}$,$\vec{j}$,$\vec{k}$ 分别为 x,y,z 三个方向的单位向量。

MATLAB 中设有 gradient 函数,它是靠数值微分的,因此空间观测点应取得密一些,以获得较高的精度。

◆MATLAB 程序

```
    V = input(':','s');        % 读入字符串,例如 log(x.^2 + y.^2)
    xMax = 5;NGrid = 20;       % 绘图区从 x=-xMax 到 x= xMax,网格线数
    xPlot = linspace( -xMax,xMax,NGrid);   % 绘图取 x 值
    [x,y]=meshgrid(xPlot);                 % x,y 取同样范围,生成二维网格
    VPlot=eval(V);             % 执行输入的字符串 V(MATLAB 语句)
    [ExPlot,EyPlot] = gradient(-VPlot);    % 电场等于电位的负梯度
    clf;subplot(1,2,1),meshc(VPlot);      % 画含等高线的三维曲面
    xlabel('x');ylabel('y');zlabel('电位');
```

```
% 规定等高线图的范围及比例
subplot(1, 2, 2), axis([-xMax xMax -yMax yMax]);    % 建立第二个子图
cs = contour(x, y, VPlot);                           % 画等高线，cs 是等高线值
clabel(cs); hold on;                                 % 在等高线图上加上编号
% 在等高线图上加上电场方向
quiver(x, y, ExPlot, EyPlot);                        % 画电场 E 的箭头图
xlabel('x'); ylabel('y'); hold off;
```

◆程序运行结果

在输入电位方程 V(x, y) = log(x.^2 + y.^2)时，得出图 6-11(*a*)所示的电位分布曲面和图 6-11(*b*)所示的电场分布向量图。

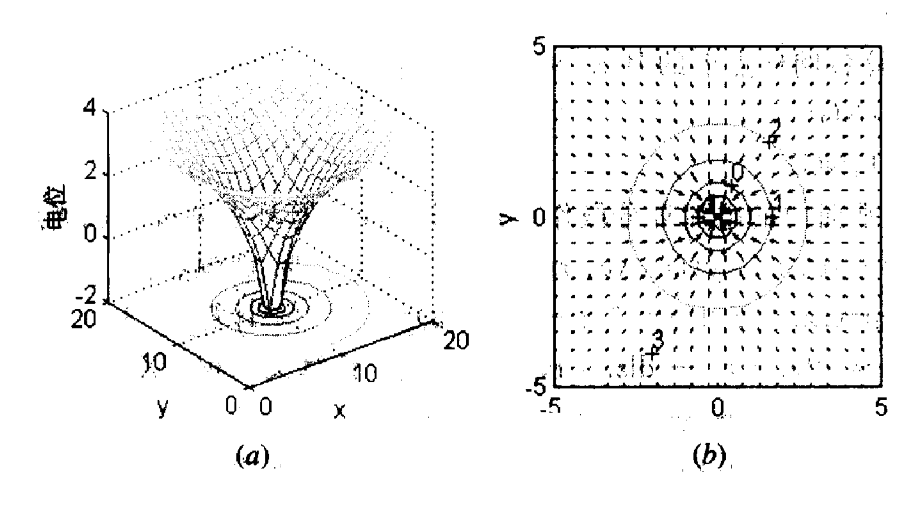


图 6-11 V(x, y) = log(x.^2 + y.^2)的电位三维立体图和等位线及电场分布图
(*a*) 电位三维立体图；(*b*) 等位线及电场分布图

6.5 恒稳磁场

【例 6-5-1】 用毕奥—萨伐定律计算电流环产生的磁场。

解：

◆建模

载流导线产生磁场的基本规律为：任一电流元 $I\,\vec{dl}$在空间任一点 P 处所产生的磁感应强度$\vec{dB}$为下列向量叉乘积，即

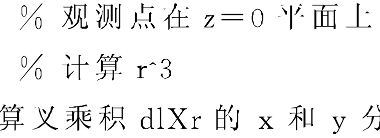
$$\vec{dB}=\frac{\mu_0}{4\pi}\cdot\frac{I\,\vec{dl}\times\vec{r}}{r^3}$$

$\vec{r}$ 为电流元到 P 点的矢径，$\vec{dl}$为导线元的长度矢量。P 点的总磁场可沿载流导体全长积分各段产生的磁场来求得。

◆MATLAB 程序

初始化，输入环半径 Rh，环电流 I0，真空导磁率 mu0 = 4 * pi * 1e-7 等的语句略

```
x=linspace(-3, 3, 20); y=x;        % 确定观测点的 x, y 坐标数组
Nh = 20;                           % 电流环分段数
  % 计算每段的端点，环在 x=0 平面上，其坐标 x1, x2 均为零
  theta0 = linspace(0, 2 * pi, Nh+1);    % 环的圆周角分段
  theta1 = theta0(1: Nh);
  y1 = Rh * cos(theta1);             % 环各段向量的起点坐标 y1, z1
  z1 = Rh * sin(theta1);
  theta2 = theta0(2: Nh+1);          % 注意 theta1 和 theta2 的差别
  y2 = Rh * cos(theta2);             % 环各段向量的终点坐标 y2, z2
  z2 = Rh * sin(theta2);
  dlx=0; dly = y2-y1; dlz = z2-z1;       % 计算环各段向量 dl 的三个长度分量
  xc=0;yc=(y2+y1)/2;zc=(z2+z1)/2; % 计算环各段向量中点的三个坐标分量
% 循环计算各网格点上的 B(x, y)值
   for i=1:NGy
    for j=1:NGx
  % 对 yz 平面内的电流环分段作元素群运算，先算环上某段与观测点之间的向量
r
  rx=x(j)-xc; ry=y(i)-yc; rz=0-zc;          % 观测点在 z=0 平面上
   r3 = sqrt(rx.^2 + ry.^2 + rz.^2).^3;      % 计算 r^3
   dlXr_x = dly. * rz - dlz. * ry;        % 计算叉乘积 dlXr 的 x 和 y 分量
   dlXr_y = dlz. * rx - dlx. * rz;
  Bx(i, j) = sum(C0 * dlXr_x. /r3);       % 把环各段产生的磁场分量累加
  By(i, j) = sum(C0 * dlXr_y. /r3);
 end
end
% 用 quiver 画磁场向量图
clf; quiver(x, y, Bx, By);
```



图形标注语句及在图上画出圆环位置的语句略

◆程序运行结果

运行此程序所得结果如图 6－12 所示，读者可改变电流环的直径来分析其影响，也可加上显示各点磁场强度的语句来分析其强度的分布。

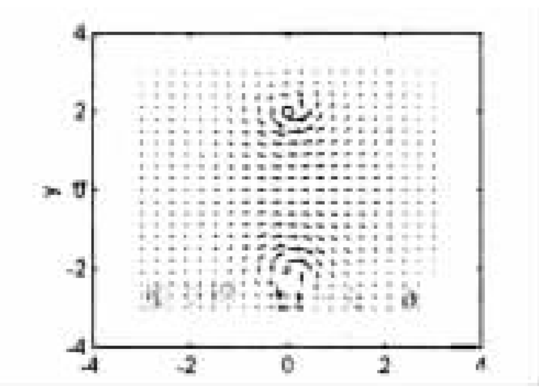


图 6－12　电流环产生的磁场分布图

【例 6－5－2】 两个平行电流环之间截面上磁场分布的计算——亥姆霍兹线圈的验证。

一对相同的共轴载流圆线圈，当它们的间距正好等于线圈半径时，称之为亥姆霍兹线圈。计算表明，亥姆霍兹线圈轴线附近的磁场的大小十分均匀，而且都沿 x 方向。本题要求对这一论断进行验证。

解：

◆建模

本题的计算模型与上例相同，只是把观测区域取在两线圈之间的小范围内。B 生成的线圈左边的磁场就等于 A 线圈的左边磁场，因此，A、B 两线圈在中间部分的合成磁场等于 A 线圈的右磁场与其左磁场平移 Rh 后的和。因此，只要观测 A 线圈的左右区间 x=[-Rh，Rh] 内的磁场就够了，如图 6-13 所示。

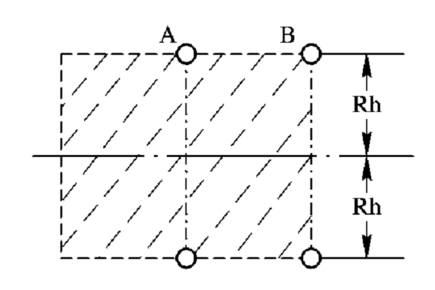


图 6-13　亥姆霍兹线圈观测区

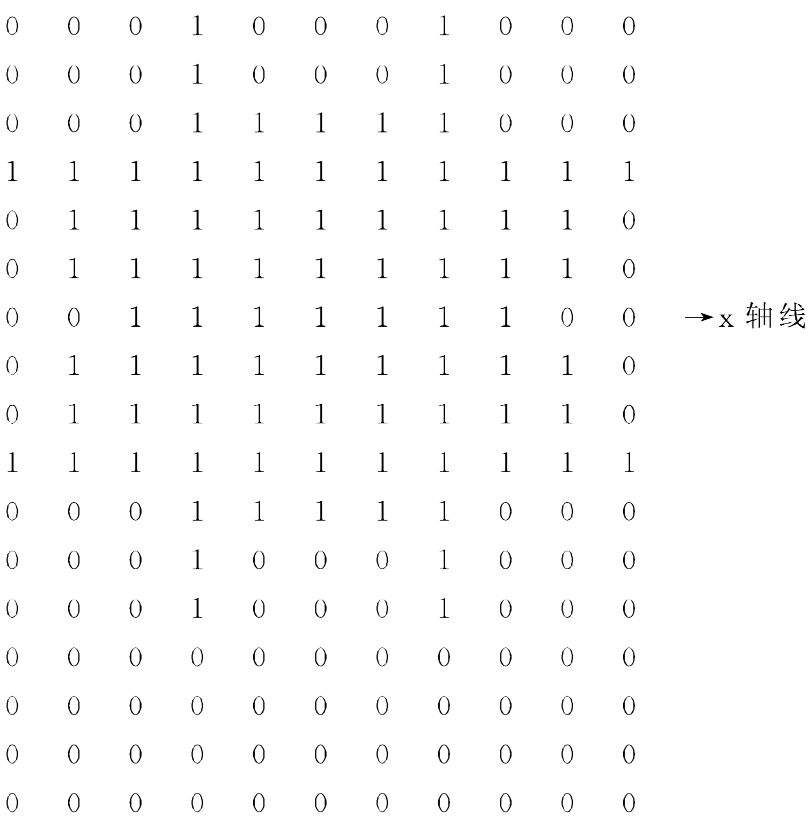
◆MATLAB 程序

```
clear all;    % 初始化(给定环半径，电流，图形)
mu0 = 4 * pi * 1e-7;        % 真空导磁率(T * m/A)
I0 = 5.0; Rh=1;             % 这两个常数不影响结果
C0 = mu0/(4 * pi) * I0;  % 归并常数
% 下面三行输入语句与上题不同，观测范围 x 取[-Rh，Rh]，即线圈的左右都取，
% 因为以后要把 A 线圈的右磁场和 B 线圈的左磁场相加
NGx =21; NGy = 21;           % 设定观测点网格数
x=linspace(-Rh, Rh, NGx);% 设定观测点范围及数组
y=linspace(-Rh, Rh, NGy);% y 也取[-Rh，Rh]
主程序段同例 6-5-1(从 Nh=20 到最后一个 end)
Bax=Bx(:,11:21)+Bx(:, 1:11); % 把 x<0 区域内的磁场平移，叠加到 x>0 区域
Bay=By(:, 11: 21)+By(:, 1: 11);
subplot(1, 2, 1),
mesh(x(11: 21), y, Bax); xlabel('x'); ylabel('y');   % 画出其 Bx 分布三维图
subplot(1, 2, 2),
plot(y, Bax), grid, xlabel('y'); ylabel('Bx');
```

◆运行结果

运行结果如图 6-14 所示。可以看出，在亥姆霍兹线圈的两个线圈之间的轴线附近，有相当大的一个区域内，x 方向的磁场强度 Bx 是非常均匀的。用相仿的语句也不难得知，该区域内的 y 方向的磁场强度 By 近似为零，这留给读者自行证明。为了看出这个区域的大小和形状，可以用多种方法，这里用最简单的一种非图形方法，键入

```
[q]=abs((Bax-Bax(11, 6))/Bax(11, 6))<0.02   %磁场相对误差<2%的点
q = 0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0
```



```
0  0  0  1  0  0  0  1  0  0  0
0  0  0  1  0  0  0  1  0  0  0
0  0  0  1  1  1  1  1  0  0  0
1  1  1  1  1  1  1  1  1  1  1
0  1  1  1  1  1  1  1  1  1  0
0  1  1  1  1  1  1  1  1  1  0
0  0  1  1  1  1  1  1  1  0  0    →x 轴线
0  1  1  1  1  1  1  1  1  1  0
0  1  1  1  1  1  1  1  1  1  0
1  1  1  1  1  1  1  1  1  1  1
0  0  0  1  1  1  1  1  0  0  0
0  0  0  1  0  0  0  1  0  0  0
0  0  0  1  0  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0
```

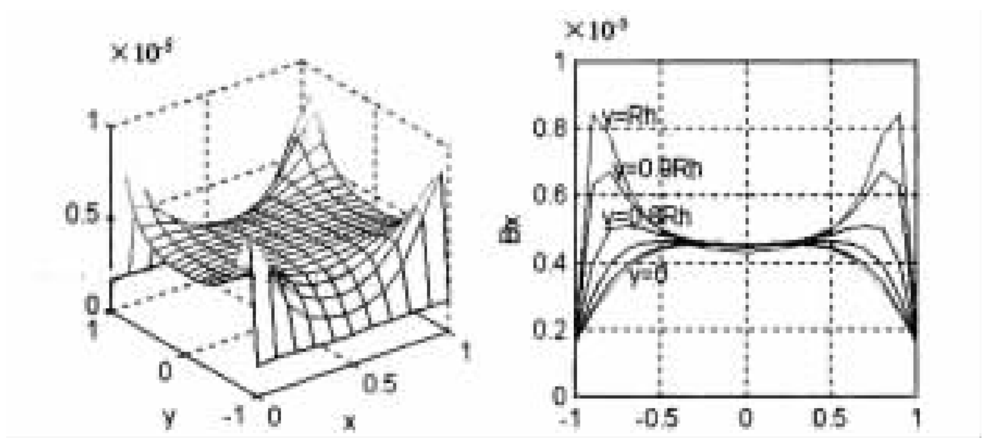


图 6－14　亥姆霍兹线圈轴线附近 Bx 按 x，y 的网格曲面(左)
和沿 y 向的分布图(右)

在图 6－14 中各行和各列的间距都是 0.05Rh，21 行 11 列分别表示了两个线圈间的 xy 截面，q＝1 处表明该点处满足上述磁场均匀性条件。应该注意，满足磁场均匀性条件的空间区域应该是以 x 为轴线的回转体。

同样键入 p ＝ abs((Bay)/Bax(11，6))<0.02，可找到 Bay 小于中心点 Bax(11,6)的 2％ 的区域。

用 format＋显示 q 和 p 能得到更紧凑的打印格式。用图形方法能得到更美观的表现，不过编程更复杂一些。

6.6　振　动　与　波

【例 6－6－1】　振动的合成及拍频现象。

分别输入两个正弦波的振幅、相位及频率，观察其合成的结果，特别是当两个信号的频率接近时产生的拍频现象。

解：

◆ 建模

两个同方向的振动 $y_1=a_1\sin(\omega_1 t+\varphi_1)$和 $y_2=a_2\sin(\omega_2 t+\varphi_2)$相加，得

$$y=y_1+y_2=a_1\sin(\omega_1 t+\varphi_1)+a_2\sin(\omega_2 t+\varphi_2)$$

用三角函数关系，可求出

$$y=(a_1+a_2)\sin\left(\frac{\omega_1+\omega_2}{2}t+\frac{\varphi_1+\varphi_2}{2}\right)\cos\left(\frac{\omega_1-\omega_2}{2}t+\frac{\varphi_1-\varphi_2}{2}\right)$$
$$+(a_1-a_2)\sin\left(\frac{\omega_1-\omega_2}{2}t+\frac{\varphi_1-\varphi_2}{2}\right)\cos\left(\frac{\omega_1+\omega_2}{2}t+\frac{\varphi_1+\varphi_2}{2}\right)$$

当 ω_1 和 ω_2 很接近时，$\frac{\omega_1-\omega_2}{2}$成为一个很低的频率，称为拍频，用 MATLAB 程序得到的图形和声音中可以很清楚地看出拍频现象。

◆MATLAB 程序

```
t=0:0.001:10;            % 给出时间轴上 10 s，分 10 000 个点
% 输入两组信号的振幅和频率
a1=input('振幅 1='); w1=input('频率 1=');
a2=input('振幅 2='); w2=input('频率 2=');
y1=a1 * sin(w1 * t);     % 生成两个正弦波
y2=a2 * sin(w2 * t);
y=y1+y2;                 % 将两个波叠加
subplot(3,1,1),plot(t,y1),ylabel('y1')    % 画出曲线
subplot(3,1,2),plot(t,y2),ylabel('y2')
subplot(3,1,3),plot(t,y),ylabel('y'),xlabel('t')
pause,sound(y1);pause(2),     % 产生声音
sound(y2);pause(2),sound(y),pause
subplot(1,1,1)           % 绘图复原
```

◆程序运行结果

键入

a1=1.2；w1=300

a2=1.8；w2=310

运行的结果如图 6－15 所示。由于这两个频率非常接近，因此产生了差拍频率。如有声卡和音箱，也能听到这个拍频。

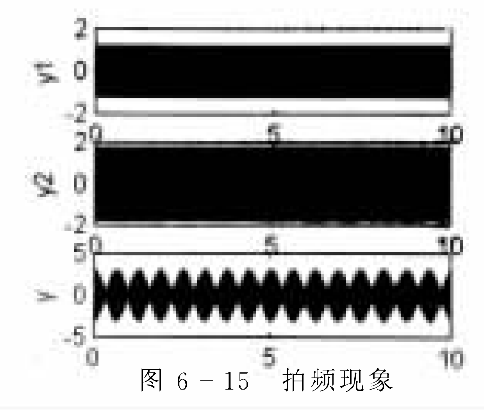


图 6－15　拍频现象

【例 6－6－2】　多普勒效应的验证。设声源从 500 m 外以 50 m/s 的速度对听者直线驶来，其轨迹与听者的最小垂直距离为 $y_0=20$ m，参看图 6－16，声源的角频率为 1000 rad/s，试求听者接收到的信号波形方程并生成其相应的声音。

解：

◆建模

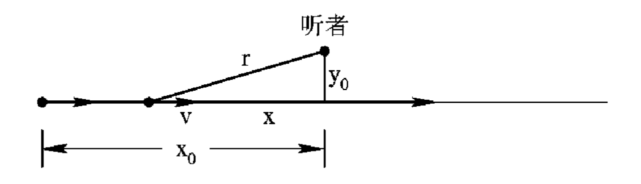


图 6－16 声源运动的几何关系

设声源发出的信号为 f(t)，传到听者处，被听者接收的信号经历了声音传播的延迟，延迟时间为

$$\Delta t = \frac{r}{c}$$

其中，c 为音速，r 为声源与听者之间的距离，故接收的信号形式为(不考虑声波的传输衰减)

$$f_1(t) = f(t - \frac{r}{c})$$

只要给出 f(t)及 r 随 t 变化的关系，即可求得 $f_1(t)$并将它恢复为声音信号。

◆MATLAB 程序

```
x0=500; v=60; y0=30;                     % 设定声源运动参数
c=330; w=1000;                           % 音速和频率
t=0: 0.001: 30;                          % 设定时间数组
r=sqrt((x0-v*t).^2+y0.^2);               % 计算声源与听者距离
t1=t-r/c;                                % 经距离迟延后听者的等效时间
u=sin(w*t)+sin(1.1*w*t);                 % 声源发出的信号设为两种频率的合成
u1=sin(w*t1)+sin(1.1*w*t1);              % 听者接受到的信号
sound(u); pause(5); sound(u1);           % 将原信号和接受到的信号恢复为声音
```

◆程序运行结果

打开计算机的声音系统，运行此程序将会听到类似于火车汽笛的声音。第一声是火车静止时的汽笛声，第二声是本题中静止的听者听到的运动火车的汽笛声，它的频率先高于原来的汽笛声，后低于原来的汽笛声。程序中两个 sound 语句之间加的 pause(暂停)语句是不可少的，而且暂停的时间要足够长，以便再打开声音系统，这个量与计算机硬件有关，在作者的计算机上，这个数至少要取 5。

6.7 光 学

【例 6－7－1】 两点(双缝)光干涉图案。

单色光通过两个窄缝射向屏幕，相当于位置不同的两个同频同相光源向屏幕照射的叠合，由于到达屏幕各点的距离(光程)不同引起相位差，如图 6－17 所示，叠合的结果是在有的点加强，在有的点抵消，造成干涉现象。考虑到纯粹的单色光不易获得，通常都有一定的光谱宽度，这种光的非单色性对光的干涉会产生何种效应，要求用 MATLAB 计算并

仿真这一问题。

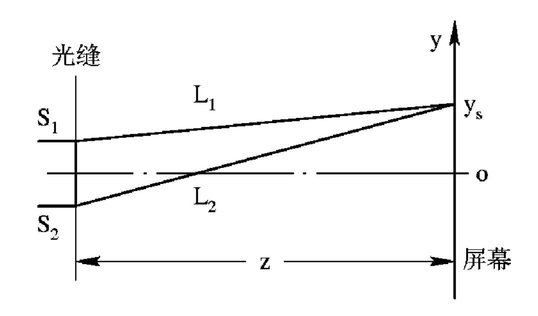


图 6-17 双缝干涉的示意图

解：

◆建模

考虑两个相干光源到屏幕上任意点的距离差引起的相位差

$$L_1=\sqrt{(y_s-\frac{d}{2})^2+z^2},\quad L_2=\sqrt{(y_s+\frac{d}{2})^2+z^2}$$

则光程差为

$$\Delta L=L_1-L_2$$

将 ΔL 除以波长 λ，并乘以 2π，得到相位差 $\varphi=2\pi\cdot\frac{\Delta L}{\lambda}$。设两束相干光在屏幕上产生的幅度相同，均为 A_0，则夹角为 φ 的两个向量 A_0 的合成向量的幅度为

$$A=2A_0\cos(\varphi/2)$$

光强 B 正比于振幅的平方，故有

$$B=4B_0\cos^2(\varphi/2)$$

根据这些关系式，可以编写出计算屏幕上各点光强的程序，本书中为 ex671.m。

考虑到光的非单色性对干涉条纹的影响，将使问题更为复杂，此时波长将不是常数，必须对不同波长的光作分类处理再叠加起来。假定光源的光谱宽度为中心波长的±10%，并且在该区域内均匀分布。近似取 11 根谱线，相位差的计算式求出的将是对不同谱线的 11 个不同相位。计算光强时应把这 11 根谱线产生的光强叠加并取平均值，即

$$\varphi_k=2\pi\frac{\Delta L}{\lambda_k}$$

$$B=\sum_{k=1}^{11}\frac{4\cos^2\left(\frac{\varphi_k}{2}\right)}{11}\cdot B_0$$

◆MATLAB 程序(在本书附盘上将分成两个程序 ex671.m 和 ex671a.m)

输入波长 Lambda=500 nm，光缝距离 d=2 mm，光栅到屏幕距离 z=1 m

```
yMax = 5 * Lambda * Z/d; xs = yMax;     % 设定图案的 y，x 向范围
Ny=101; ys = linspace(-yMax, yMax, Ny);    % y 方向分成 101 点
for i=1: Ny              % 对屏上全部点进行循环计算
  % 计算第一个和第二个光源到屏上各点的距离
  L1 = sqrt((ys(i)-d/2).^2 + z^2 );
```

```
    L2 = sqrt((ys(i)+d/2).^2 + z^2 );
    Phi = 2 * pi * (L2-L1)/Lambda;          % 从距离差计算相位差
    B(i,:) = 4 * cos(Phi/2).^2;             % 计算该点光强(设两束光强相同)
    % 若考虑光谱的非单色性，把前两句改为后四句(可将前两句加"%"号，
    % 而取消后
    % 四句前的"%"号)，由此构成程序 ex671a.m
    % Nl=11; dL=linspace(-0.1, 0.1, Nl);     % 设光谱相对宽度±10%
    % Lambda1=Lambda * (1+dL');              % 分 11 根谱线，波长为一个数组
    % Phi1 = 2 * pi * (L2-L1)./Lambda1;       % 从距离差计算各波长的相位差
    % B(i,:) = sum(4 * cos(Phi1/2).^2)/Nl;   % 叠加各波长影响计算光强
end
clf; figure(gcf);              % 清图形窗，将它移到前面，准备绘图
NCLevels = 255;                % 确定用的灰度等级为 255 级
% 定标：使最大光强(4.0)对应于最大灰度级(白色)
Br = (B/4.0) * NCLevels;
subplot(1, 2, 1), image(xs, ys, Br);       % 画图像
colormap(gray(NCLevels));                  % 用灰度级颜色图
subplot(1, 2, 2), plot(B(:), ys)           % 画出沿 y 向的光强变化曲线
```

◆程序运行结果

分别运行 ex671 和 ex671a 两个程序所得的屏幕光强图像见图 6－18，可以看出，光的非单色性导致干涉现象的减弱。光谱很宽的光将不能形成干涉。

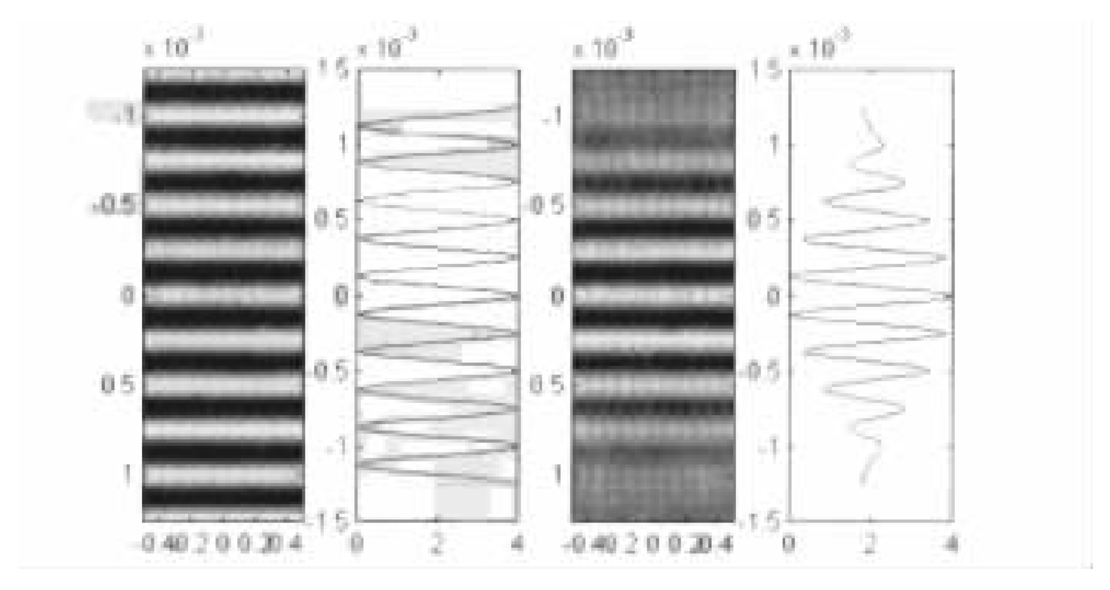


图 6－18 双缝干涉条纹及光强分布

左图为严格单色光时的结果，右图为非单色光△Lambda=±0.1 * Lambda 时的结果)

【例 6－7－2】 用 MATLAB 程序来计算演示光的单缝衍射现象。

解：

◆模型

把单色平行光通过的光缝当作 N 点干涉来计算，单缝衍射的几何关系如图 6－19

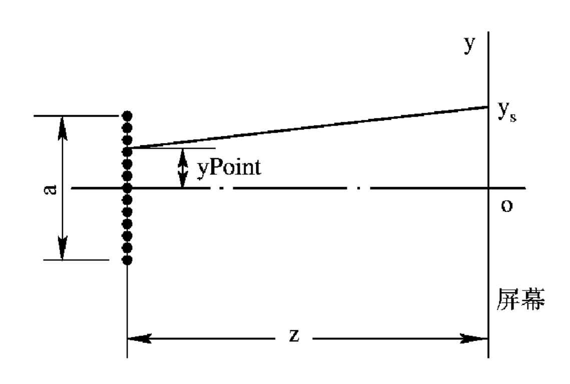


图 6－19　单缝衍射几何关系

所示。

把单缝看做一排等间隔光源，共 NPoint 个光源分布在－a/2～＋a/2 区间内，则屏幕上任一点 y_s 处的光强为这 NPoint 个光源照射结果的合成。

设某光源的 y 坐标为 yPoint，则它到 y_s 的路程为

$$L=\sqrt{(y_s-yPoint)^2+z^2}$$

◆MATLAB 程序

输入波长 Lambda＝500 nm，缝宽 a＝0.2，1，2 mm 等三种情况，距离 z＝1 m 的语句

```
ymax = 3 * Lambda * Z/a;                    % 屏幕范围(沿 y 向)
Ny = 51;                                    % 屏幕上的点数(沿 y 向)
ys = linspace(-ymax, ymax, Ny);
NPoints = 51;                               % 缝上的点数(沿 y 向)
yPoint = linspace(-a/2, a/2, NPoints);%     把缝上的点数设成数组
for j=1: Ny                                 % 对屏幕上 y 向各点作循环
% 对光缝中各点作循环，计算缝点到屏幕位置的距离
  L = sqrt((ys(j)-yPoint).^2 + z^2 );      % L 是一个数组
  Phi = 2 * pi. * (L-z)./Lambda;          % 对于屏幕中心的相位差，也是一个数组
  % 求每个分量的累加和，假定各光源在 ys 处产生的光强相同，只是相位不同
  SumCos = sum(cos(Phi));                   % 数组求和
  SumSin = sum(sin(Phi));
 % 求屏幕上的归一化光强;
 B(j) = (SumCos^2 + SumSin^2)/NPoints^2;
 end
clf, plot(ys, B, '*', ys, B); grid;         % 屏幕上光强与位置的关系曲线
```

图形标注语句

◆程序运行结果

缝宽为 0.2 mm，1 mm 和 2 mm 等三种情况的程序运行结果如图 6－20 所示。三种情况统称费涅耳衍射，只有最左边的情况符合夫琅和费衍射的条件（也称远场条件），即

$$\frac{\pi a^2}{(4\lambda z)}\ll 1$$

这个现象也适用于研究电磁波的发射。天线的设计和测量都要用这个概念。天线探测

目标时通常符合夫琅和费衍射的条件，形成天线的远场波瓣。但在天线测量时却希望在近处测量，这时就不符合远场的条件，于是要建立远场和近场之间的转换关系，MATLAB程序可以发挥作用。

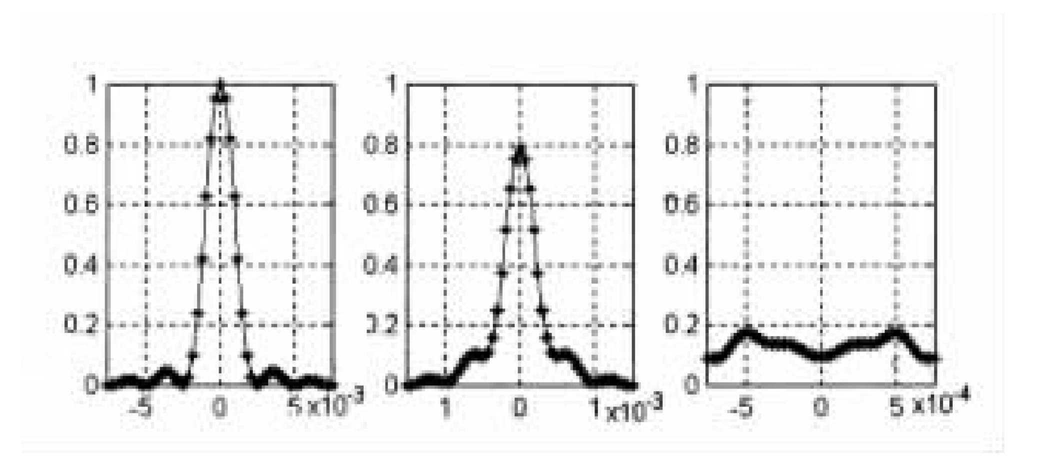


图 6 - 20　缝宽 a 为 0.2 mm，1 mm 和 2 mm 等三种情况所得衍射光强曲线(左为夫琅和费衍射)

第 7 章 在力学、机械中的应用举例

7.1 理论力学

【例 7-1-1】 给定由 N 个力 $\vec{F}_i(i=1,2,\cdots,N)$ 组成的平面任意力系，求其合力。

解：

◆建模

本程序可用来对平面任意力系作简化，得出一个合力。求合力的过程可分成两步。

第一步：向任意给定点 o 简化，得到一个主矢 $\vec{F}_o$ 和一个主矩 M_o，即

$$\vec{F}_o = \sum_{i=1}^{N} \vec{F}_i [F_{ox}, F_{oy}]$$

$$M_o = \sum_{i=1}^{N} (\vec{r}_i - \vec{r}_o) \times \vec{F}_i = \sum_{i=1}^{N} (x_i - x_o) F_{yi} - (y_i - y_o) F_{xi}$$

式中，$\vec{r}_i$ 是 $\vec{F}_i$ 作用点的矢径；$\vec{r}_o$ 是 o 点的矢径。

第二步：将此主矢和主矩向 T 点转移，使其力矩 M_t 为零，成为一个合力 F_t。令

$$M_t = (\vec{r}_o - \vec{r}_t) \times \vec{F}_o + M_o = 0$$

注意向量可以用数组表示，用 1×2 数组来表示平面向量，此式可化为

$$[r_o - r_t] \cdot \begin{bmatrix} F_{oy} \\ F_{ox} \end{bmatrix} = -M_o$$

用 MATLAB 的右除符号，可以得到合力作用点 T 的坐标 r_t 为

$$r_t = M_o / \begin{bmatrix} F_{oy} \\ F_{ox} \end{bmatrix} + r_o$$

式中，r_t 和 r_o 都是 1×2 的数组，可由此式解出 r_t。

◆MATLAB 程序

```
clear, N=input('输入力的数目 N=')        % 输入力系中各力的数据
for i=1: N
   i, F(i,: )=input('力 F(i)的 x, y 两个分量[Fx(i), Fy(i) ] = ');
   r(i,: )=input('力 F(i)的一个作用点的坐标 r(i)= [rx, ry ] = ');
end
ro=input('简化中心 ro 的坐标 ro= [xo, yo ] = ');    % 输入简化中心的数据
Fo=sum(F),                              % 求主矢 Fo=[Fox, Foy]
for i=1: N                              % 计算各力对 ro 点的力矩
```

```
    M(i)=F(i, 2)*(r(i, 1)-ro(1))-F(i, 1)*(r(i, 2)-ro(2));
end
Mo=sum(M)                          % 相加求主矩
rt = Mo/[Fo(2);-Fo(1)] + ro        % 求合力作用点的坐标
```

◆程序运行结果

最后一条语句从一个方程要求出两个未知数 rt(1)和 rt(2)，这是一个不定方程，事实上合力作用线将通过平面上的无数点，程序中用矩阵右除的方法将给出的无数个解之一，即 rt-ro 中有一个分量是零的那个解。

运行此程序，输入

N=3，F(1，:)=[2，3]，r(1，:)=[-1，0]，

F(2，:)=[-4，7]，r(2，:)=[1，-2]，

F(3，:)=[3，-4]，r(3，:)=[1，2]，

又设简化中心的坐标 ro=[-1，-1]，答案为

Fo =[1   6]，Mo = -9(即 x 方向分力为 1，y 方向分力为 6)

rt =[-2.5000   -1.0000](合力作用线通过的某一点坐标)

【例 7-1-2】 求图 7-1 所示杆系的支撑反力 N_a，N_b，N_c。

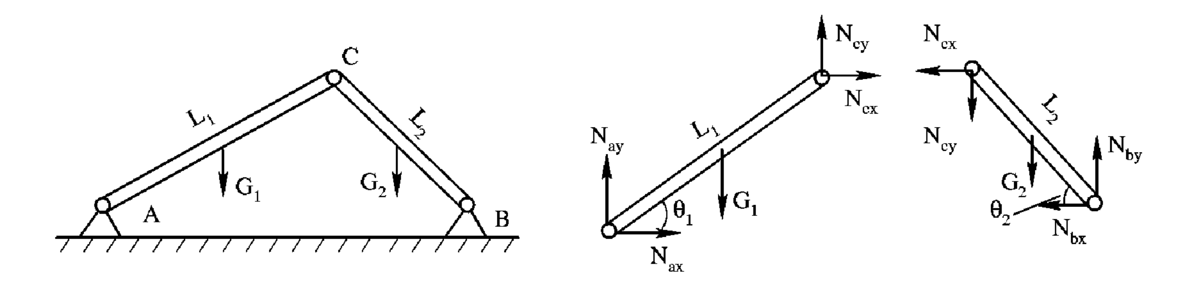


图 7-1 杆系结构及受力图

设已知：$G_1=200$；$G_2=100$；$L_1=2$；$L_2=\sqrt{2}$；$\theta_1=30°$；$\theta_2=45°$。

解：

◆建模

画出杆 1 和杆 2 的受力图(如图 7-1 所示)，列出方程。

对杆 1：

$$\sum X = 0 \qquad N_{ax} + N_{cy} = 0$$

$$\sum Y = 0 \qquad N_{ay} + N_{cy} - G_1 = 0$$

$$\sum M_a = 0 \qquad N_{cy} L_1 \cos\theta_1 + N_{cy} L_1 \sin\theta_1 - G_1 \frac{L_1}{2} \cos\theta_1 = 0$$

对杆 2：

$$\sum X = 0 \qquad N_{bx} - N_{cx} = 0$$

$$\sum Y = 0 \qquad N_{by} - N_{cy} - G_2 = 0$$

$$\sum M_b = 0 \qquad N_{cy} L_2 \cos\theta_2 + N_{cx} L_2 \sin\theta_2 + \frac{1}{2} G_2 L_2 \cos\theta_2 = 0$$

这是一个包含六个未知数 N_{ax}、N_{ay}、N_{bx}、N_{by}、N_{cx} 和 N_{cy} 的六个线性代数方程，要解这个方程组，通常是要寻找简化的步骤，但用了 MATLAB 工具，也可以不简化，把方程组写成矩阵形式 AX＝B，用矩阵除法 X＝A　B 直接来解。在本题中，X 和 B 都是 6×1 列向量，而 A 是 6×6 方阵。

在编写程序时，尽量用文字变量，先输入已知条件，在程序开始处给它们赋值，这样得出的程序具有一定的普遍性，若要修改参数，只需修改头几行的数据即可。

◆MATLAB 程序

```
G1＝200；G2＝100；L1＝ 2；L2 ＝ sqrt (2)；      % 给原始参数赋值
theta1 ＝30 * pi/180；theta2 ＝ 45 * pi/180；     % 将度化为弧度
% 设 X＝[ Nax；Nay；Nbx；Nby；Ncx；Ncy ]，则系数矩阵 A，B 可写成下式
A＝[1,0,0,0,1,0;0,1,0,0,0,1;0,0,0,0,－sin(theta1),cos(theta1);...
    0,0,1,0,－1,0;0,0,0,1,0,－1;0,0,0,0,sin(theta2),cos(theta2)]
B＝[0；G1；G1/2 * cos(theta1)；0；G2；－G2/2 * cos(theta2)]
X ＝ A  B；                  %用左除求解线性方程组
disp('Nax，Nay，Nbx，Nby，Ncx，Ncy')        % 显示结果
disp(X')
```

◆程序运行结果

```
A ＝ 1.0000        0         0         0      1.0000        0
        0      1.0000        0         0         0      1.0000
        0         0          0         0     －0.5000    0.8660
        0         0       1.0000       0     －1.0000       0
        0         0          0      1.0000       0     －1.0000
        0         0          0         0      0.7071    0.7071
B ＝ [ 0    200.0000    86.6025     0    100.0000    －35.3553 ]'
      Nax        Nay        Nbx        Nby        Ncx        Ncy
    95.0962   154.9038  －95.0962   145.0962  －95.0962   45.0962
```

【例 7－1－3】 设导弹 M 速度分别为 v_m＝1000 m/s 和 800 m/s，其速度向量始终对准速度为 v_t＝500 m/s 的直线飞行目标 T，发射点在目标运动方向的左(4000 m)前(3000 m)方，如图7－2 所示，试求导弹轨迹及其加速度。

解：

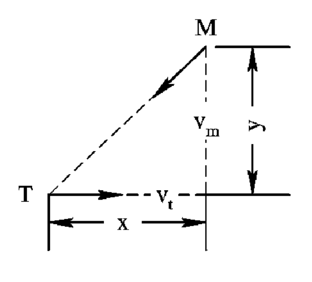


图 7－2　导弹攻击目标的运动

◆建模

在与目标固连的等速直线运动坐标(惯性坐标系)中列写动点 M 的方程。因动坐标与目标 T 固连，牵连速度 $\vec{v}_e=\vec{v}_t$，动点为 M，它的绝对速度 $\vec{v}_a=\vec{v}_m$。由速度合成定理，相对速度 $\vec{v}_r=\vec{v}_a-\vec{v}_e=\vec{v}_m-\vec{v}_t$，列出其在 x，y 两方向的投影，得

$$v_{rx}=\frac{dx}{dt}=-v_m\frac{x}{\sqrt{x^2+y^2}}-v_t$$

$$v_{ry}=\frac{dy}{dt}=-v_m\frac{y}{\sqrt{x^2+y^2}}$$

求其积分，即可求得其轨迹 $x=x(t)$，$y=y(t)$。

◆MATLAB 程序(ex713.m)

MATLAB 数值积分要求把导数方程单独列写为一个函数程序，故其 MATLAB 程序由主程序和一个求导数的函数程序构成。由于数值积分的步长是 MATLAB 按精度自动选取的，其间隔可变，因此 dt 要用数组表示。

主程序 ex713.m。

```
global vt vm
vt=input('vt='); vm=input('vm=');        % 输入主程序及函数程序共用的参数
z0=input('[x0; y0]=');                   % 输入数值积分函数需要的参数
tspan=input('tspan=[t0, tfinal]=');      % 输入数值积分函数需要的参数
[t, z] = ode23('ex713f', tspan, z0);     % 进行数值积分
plot(z(:, 1), z(:, 2));                  % 绘图
% 在惯性坐标中，M 点位置的导数是相对速度，而其二次导数则为绝对加速度
dt=diff(t); Ldt=length(dt);   % 为了求导数，先求各时刻处 t 的增量
x=z(:, 1); y=z(:, 2);                    % 把 z 写成 x，y 两个分量形式
vx=diff(z(:, 1))./dt; vy=diff(z(:, 2))./dt; % 注意每差分一次序列长度少 1
wx=diff(vx)./dt(1: Ldt-1); wy=diff(vy)./dt(1: Ldt-1);   % 求二次导数
[t(2: Ldt), x(2: Ldt), y(2: Ldt), wx, wy]     % 显示数据
```

下面是函数程序，应存成一个文件 ex713f.m。

```
function zprime=ex713f(t, z)
global vt vm
zprime=[0; 0]; % 给出 t0 之前 zprime 初值(缺了这个值 ode23 积分会无法进行)
zprime(1)=-vt-vm * z(1)/sqrt(z(1)^2+z(2)^2);
zprime(2)=-vm * z(2)/sqrt(z(1)^2+z(2)^2);
% 也可换成矩阵语句: zprime=-vt * [1; 0]-vm * z/sqrt(z(1)^2+z(2)^2)
```

◆程序运行结果

把这两个程序均存入 MATLAB 的搜索路径上。运行主程序并输入以下参数：

```
vt=500; vm=1000
[x0; y0]=[3000; 4000]
tspan=[t0, tfinal]=[0, 4.5]
```

得出图形如图 7-3 所示，数据如表 7-1 所示，为节省篇幅，表中省略了一些数据。

表　7－1

t	x	y	wx	wy
0	3000.0	4000 0		
0.2	2761.6	3824 3	110.2	－78.3
1.1	1816.8	3072 9	104.4	－61.4
2.0	958.2	2271 7	173.7	－72.3
2.9	245.0	1413 0	246.1	－46.7
3.6	－121.3	713 5	721.3	128.6
4.1	－190.0	268.7	874.5	603.7
4.4	－104.5	47.7	731.2	1588.6
4.5	－80.7	27.1		

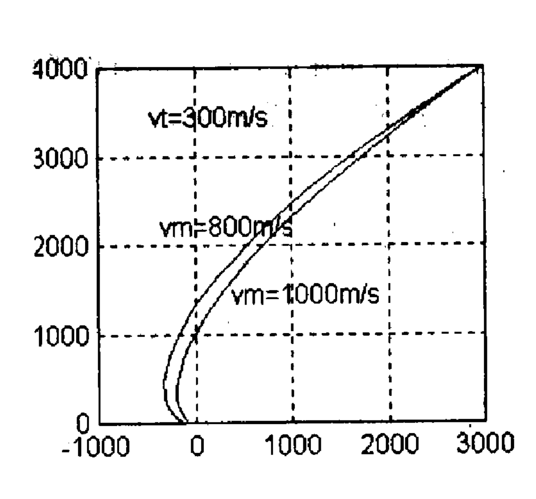


图 7－3　导弹跟踪目标时的相对轨迹

注意：在给定 tfinal 时，必须使它小于遭遇点的值，否则数字积分会进入死循环而得不出结果。读者可以思考，能否修改程序，使它能自动寻找到 tfinal，并避免进入死循环。不过这就不能用现成的 ode23 函数，而要自己编写数值积分子程序才行。

将 vm 换成 800，并相应地把 tfinal 换成 6，得到的轨迹位于图 7－3 中原轨迹的左上方。

【例 7－1－4】　四连杆机构如图 7－4 所示，输入杆 L_1 的转角 $\theta_1=\omega_1 t$，$\omega_1=100$，求输出杆 L_3 转角 θ_3 随时间的变化规律，并求其角速度和角加速度。

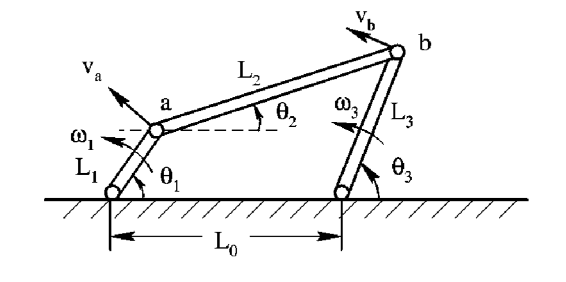


图 7－4　四连杆机构的几何关系

解：

◆建模

四连杆机构的运动方程由 X 和 Y 方向的长度关系确定为：

$$L_1\cos\theta_1+L_2\cos\theta_2-L_3\cos\theta_3-L_0=0 \tag{1}$$

$$L_1\sin\theta_1+L_2\sin\theta_2-L_3\sin\theta_3=0 \tag{2}$$

从上述两个方程中消去 θ_2，便可化成一个只包括 θ_1 和 θ_3 的方程，给定 θ_1，可求出满足此方程的 θ_3。

由(2)式得

$$\sin\theta_2=(L_3\sin\theta_3-L_1\sin\theta_1)/L_2 \tag{3}$$

将(1)式中的 $\cos\theta_2$ 代以 $\sqrt{1-\sin^2\theta_2}$，得出

$$f(\theta_1,\theta_3)=L_1\cos\theta_1+L_2\sqrt{1-\left(\frac{L_3\sin\theta_3-L_1\sin\theta_1}{L_2}\right)^2}-L_3\cos\theta_3-L_0=0 \tag{4}$$

在 θ_1 给定时，求能使 $f(\theta_3)=0$ 的 θ_3 值，然后，θ_2 就可由(3)式求得。

为了求能使 f=0 的 θ_3 值，可调用 MATLAB 中的 fzero 函数。为此，要把 $f=f(\theta_3)$ 单独定义为一个 MATLAB 函数 ex714f.m，在主程序中要调用它。为了把长度参数传给子程序，在主程序和子程序中都加了全局变量语句(global)，但全局变量容易造成程序的混乱，要特别小心，在复杂的程序中应尽量避免使用。

求得 θ_1，θ_2 和 θ_3 后，就不难根据杆 1 的角速度求出杆 3 的角速度，其方法有以下两种：

(1) 求瞬时速度，这是通常理论力学的解法，其依据就是杆 2 两端点 a 和 b 的速度沿杆长方向的分量相等，通过三角关系，有

$$L_1\omega_1\cos\left(\frac{\pi}{2}-\theta_1+\theta_2\right)=L_3\omega_3\cos\left(\theta_3-\frac{\pi}{2}-\theta_2\right)$$

从而

$$\omega_3=\frac{L_1\omega_1\cos\left(\frac{\pi}{2}-\theta_1+\theta_2\right)}{L_3\cos\left(\theta_3-\frac{\pi}{2}-\theta_2\right)}$$

由杆 2 两端点 a 和 b 的速度沿杆长垂直方向的分量之差，可以求出杆 2 的角速度

$$\omega_2=\frac{1}{L_2}\left(-L_3\sin\left(\theta_3-\frac{\pi}{2}-\theta_2\right)-L_1\omega_1\sin\left(\frac{\pi}{2}-\theta_1+\theta_2\right)\right)$$

(2) 求运动全过程的角位置、角速度和角加速度曲线，这只有借助于计算工具才能做到，因为用手工算一个点就不胜其烦，算几十个点是很难想象的。而由 MATLAB 编程调用 fzero 函数时，要求给出一个近似猜测值，若连续算几十点，前一个解就可作为后一个解的猜测值，从而带来了方便。

本书将提供 ex714a.m 和 ex714b.m 两个程序来分别表述这两种方法，它们所要调用的函数程序命名为 ex714f.m。

◆MATLAB 程序

1. 主程序 ex714a.m

```
global L0 L1 L2 L3 th1
L0=20; L1=8; L2=25; L3=20;    % 输入基线及三根杆的长度 L1, L2, L3
theta1=input('当前角 theta1= ');
theta3=input('对应于 theta1 的 theta3 近似值= ');
th1=theta1; theta3=fzero('ex714f', theta3);    % 求当前输出角 theta3
theta2 = asin(( L3 * sin(theta3)- L1 * sin(theta1))/L2);
w1=input('w1= ');
w3 = L1 * w1 * cos(pi/2-theta1+theta2)/ (L3 * cos(theta3-pi/2-theta2))
```

2. 主程序 ex714b.m

```
global L0 L1 L2 L3 th1
L0=20; L1=8; L2=25; L3=20;    % 输入基线及三根杆的长度 L1, L2, L3
w1=input('杆 1 角速度 w1= ');
theta1=linspace(0, 2 * pi, 181);    % 把杆 1 每圈分为 180 份，间隔 2°
theta3=input('对应于 theta1 最小值处的 theta3(近似估计值)= ');
```

```
dt = 2 * pi/180/w;        % 杆 1 转 2°对应的时间增量
th1=theta1(1); theta3(1)=fzero('ex714f', theta3);    % 求初始输出 theta3
for i=2:181
   th1=theta1(i);
   theta3(i)=fzero('ex714f', theta3(i-1)); % 调用 fzero 函数逐次求 theta3
end
subplot(1, 2, 1), plot(theta1, theta3), ylabel('theta3'), grid   % 画曲线
w3 = diff(theta3)/dt;     % 求杆 3 的角速度，注意求导数后数组长度小于 1
subplot(1, 2, 2), plot(theta1(2:length(theta1)), w3); grid % 画角速度曲线
```

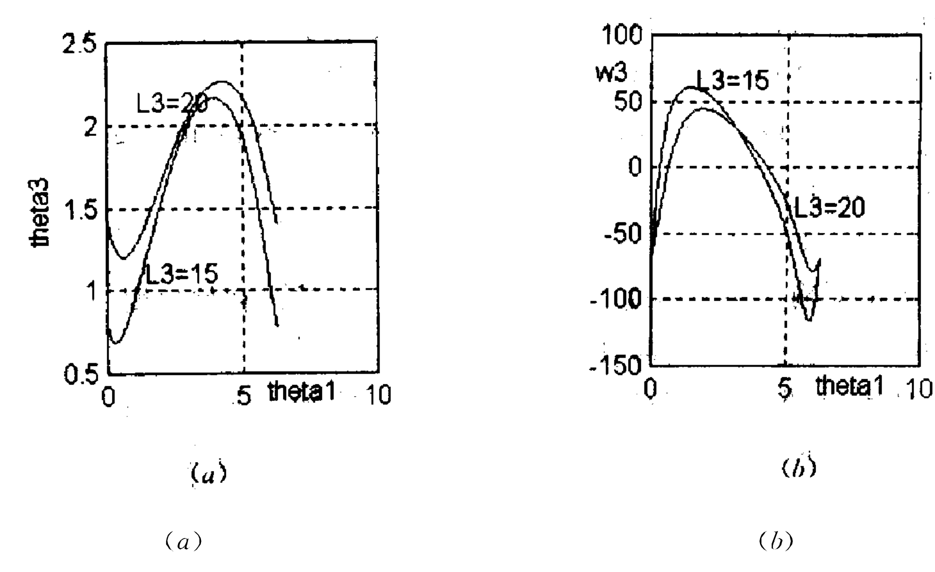
3. 子程序(函数程序)ex714f.m

```
function y=ex714f(x)
global L0 L1 L2 L3 th1
y=L1.*cos(th1)+L2*sqrt(1-(L3*sin(x)-L1*sin(th1)).^2/L2/L2)...
    -L3*cos(x)-L0;
```

在上述程序中注意 th1 是一个标量，而 theta1 在 ex714b 中是一个数组，因为函数 ex714f 中用到的是特定点的角度，是一个标量，所以不能用 theta1，引入了 th1 作为其当前值。

◆程序运行结果

在 L0=20，L1=8，L2=25，L3=20(及 15)的条件下运行 ex714b，根据提问，输入 $w_1=100$，在 theta1=0 处，设 theta3 的近似初值为 1 弧度，所得的曲线如图 7－5(*a*)所示，相应的角速度变化规律如图 7－5(*b*)所示。若运行 ex714a，其单点的数据与图7－5 一致，读者请自行检验。



(*a*)　　　　(*b*)

图 7－5　四连杆机构的输入输出角位置关系(*a*)和输出角速度(*b*)

利用 MATLAB，可以用动画来显示四连杆的运动，运行程序集中的 ex714d 就可以看到它的结果。有兴趣的读者可以打开此程序并读懂它的原理，它并不难懂。

【例 7－1－5】 考虑空气阻力时抛射体质心的飞行轨迹，设空气阻力的方向与速度向量相反，大小与速度的平方成正比。抛射体受力图如图7－6 所示。求计算质点飞行的轨迹

和距离。

解：

◆建模

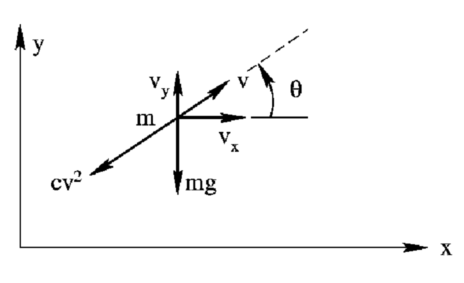


图 7－6　抛射体受力图

在例 6－2－1 中，研究过不计空气阻力的抛射体飞行轨迹，考虑空气阻力后，程序要复杂一些。因为 x 和 y 两个方向的方程会通过空气阻力互相耦合，必须联立起来求解。根据受力图7－6可列写其运动方程：

$$\frac{dx}{dt}=v_x \tag{1}$$

$$\frac{dy}{dt}=v_y \tag{2}$$

$$m\frac{dv_x}{dt}=-cv^2\cos\theta=-cv^2\cdot\frac{v_x}{v}=-cv\cdot v_x \tag{3}$$

$$m\frac{dv_y}{dt}=-cv^2\sin\theta-mg=-cv^2\frac{v_y}{v}-mg=-cvv_y-mg \tag{4}$$

式中，c 为空气阻力系数。

本来可以单独把两个速度导数的方程联立起来数值积分，再积分求位置。但在 MATLAB 中，阶次高并不会造成困难，分成两步求解，反而增加编程的工作量，所以可一次解出这个四阶方程。这里设了一个四行的向量 r=[x；y；vx；vy]来表示这四个变量。为了调用 MATLAB 的数值积分函数，必须把其运动方程组写成一个函数文件，取其文件名为 ex715f.m。它是一个四行的矩阵方程，表明系统为四阶。

◆MATLAB 程序

1. 函数程序 ex715f.m

```
function rdot=ex715f(t, r)
c = 0.01; g = 9.81; m=1;        % 给出空气阻力系数及重力加速度（m/s^2)
vm=sqrt(r(3)^2+r(4)^2);      % 速度大小
rdot = [ r(3); r(4); -c * vm * r(3)/m; -c * vm * r(4)/m- g ]; % 运动方程
```

2. 主程序 ex715.m

```
clear; y0 = 0; x0 = 0;         % 初始位置
vMag = input('输入初始速度 (m/s): ');        % 输入初始速度
vDir = input(' 输入初速方向(度): ');
tf = input('输入飞行时间(s): ');              % 输入飞行时间
vx0 = vMag * cos(vDir * (pi/180));         % 计算 x, y 方向的初始速度
vy0 = vMag * sin(vDir * (pi/180));
r0 = [0; 0; vx0; vy0];
[t, r] = ode45('ex715f', [0, tf], r0),     % 数值积分(调用函数程序 ex715f.m)
plot(r(:, 1), r(:, 2)), hold on           % 计算轨迹
```

```
% ode45 规定返回的结果中：t 是列向量，各时刻的 r 成为四列向量
% 注意下一语句的意义：找 y<0 的下标所对应的 x 的最小值，以粗略计算射程
xmax = min(r(find(r(:, 2)<0), 1))
plot([0, 150], [0, 0])                        % 画出 x 坐标线
```

◆程序运行结果

输入初始速度 (m/s)：60

输入初速方向(度)：45

输入飞行时间(s)：6.2

t	x	y	vx	vy
0	0	0	42.4264	42.4264
0.3002	11.7236	11.3046	36.0492	33.3239
1.1646	37.8640	32.1793	25.7857	16.5362
…	…	…	…	…
5.6509	111.8578	4.5714	10.0477	−22.1716
6.2000	117.0149	−8.2190	8.7531	−24.3438

xmax = 117.0149

换新的参数：

输入初始速度 (m/s)：60

输入初速方向(度)：35

输入飞行时间(s)：6

得到近似射程 xmax=123.194 6。

其轨迹如图 7－7 所示。读者可思考如何能求出射程的精确值。

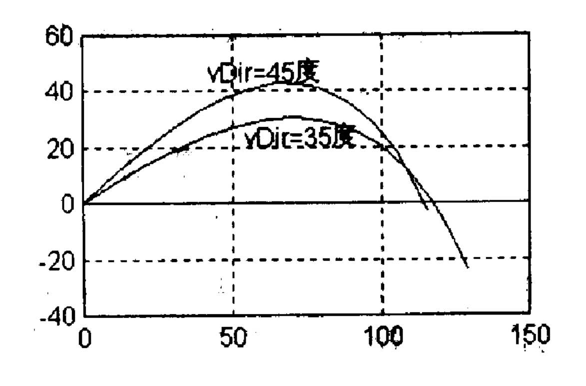


图 7－7 考虑空气阻力后的抛射体轨迹

【例 7－1－6】 给定半径为 r，重量为 Q 的均质圆柱，轴心的初始速度为 v_0，初始角速度为 ω_0，且 $v_0 > r\omega_0$，地面的摩擦系数为 f，问经过多少时间后，圆柱将无滑动地滚动，求此时圆柱轴心的速度。

解：

◆建模

圆柱受力情况如图 7－8 所示，接触面之间打滑时，摩擦力使圆柱质心减速，而使其转动加速。当圆柱触地点 C 的线速度达到 0，即 $v=\omega * r$ 时，进入纯滚动状态。

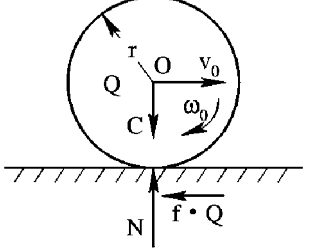


图 7－8 圆柱运动受力图

列出动力学方程

$$\sum Y = 0, \text{得 } N = Q$$

$$\sum X = m\frac{dv}{dt}, \quad \frac{Q}{g}\frac{dv}{dt} = -f \cdot Q \qquad (1)$$

$$\sum M_0 = J\frac{d\omega}{dt}, \text{因为 } J = \frac{Qr^2}{2g} \quad \text{得}$$

$$\frac{d\omega}{dt}=\frac{2fg}{r} \tag{2}$$

积分可得：

$$v=v_0-fgt \tag{3}$$

$$\omega=\omega_0+\frac{2fg}{r}t \tag{4}$$

将(3)式和(4)式联立，可求得满足 $v=r\omega$ 的时刻为 $t_1=\frac{v_0-\omega_0 r}{fg}$。

◆MATLAB 程序

```
r=1; Q=100; g=9.81;                      % 输入常数
f=0.1; v0=3; w0=2;
J = Q * r^2/2/g; F=f * Q;                % 要计算的常数
wdot = F * r/J;                          % 绕质心转动加速度方程
vdot = -F/(Q/g);                         % 质心线加速度方程
t1 = (v0-w0 * r)/(wdot * r-vdot)         % 求 t1 的方程
v = v0 + vdot * t1                       % 求 v 的方程
```

◆程序运行结果

运行此程序的结果为：

```
t1 = 0.3398,    v = 2.6667
```

即经过 0.339 8 s 后，圆柱体进入纯滚动状态，此时质心速度为 2.666 7 m/s。

7.2　材 料 力 学

【例 7-2-1】 拉压杆系的静不定问题。由 n 根杆组成的桁架结构如图 7-9 所示，受力 P 的作用，各杆截面面积分别为 A_i，材料弹性模量为 E，求各杆的轴力 Ni 及节点 C 的位移。

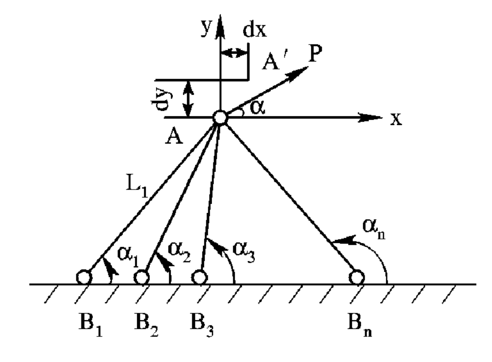


图 7-9　任意静不定杆系受力图

解：

◆建模

先列写具有普遍意义的方程，设各杆均受拉力，A 点因各杆变形而引起的 x 方向位移 Δx，y 方向位移 Δy，由几何关系，得变形方程

$$\Delta L_i = \frac{N_i L_i}{EA_i} = \Delta x\ \cos\alpha_i + \Delta y\ \sin\alpha_i \qquad (i=1,\cdots,n)$$

即

$$\frac{N_i}{K_i} - \Delta x\ \cos\alpha_i - \Delta y\ \sin\alpha_i = 0$$

其中，$K_i = \frac{EA_i}{L_i}$为杆 i 的刚度系数。

再加上两个力平衡方程

$$\sum_{i=1}^{n} N_i \cos\alpha_i = P \cos\alpha$$

$$\sum_{i=1}^{n} N_i \sin\alpha_i = P \sin\alpha$$

共有 n+2 个方程，其中包含 n 个未知力和两个待求位移 Δx 和 Δy，方程组可解。因为这又是一个线性方程组，可写成 D * X＝B 的标准形式，所以可由 MATLAB 的矩阵除法 X＝D　B 解出。

算例：设三根杆组成的桁架如图 7－10 所示，挂一重物 P＝3000 N，设 L＝2 m，各杆的截面积分别为 $A1=200\times10^{-6}\ m^2$，$A2=300\times10^{-6}\ m^2$，$A3=400\times10^{-6}\ m^2$，材料的弹性模量 $E=200\times10^{9}\ N/m^2$，求各杆受力的大小。

此时应有五个方程如下：

$$-N_1\cos\alpha_1 - N_2 - N_3\cos\alpha_3 = 0$$

$$N_1\sin\alpha_1 - N_3\sin\alpha_3 = 0$$

$$N_1/K_1 = \Delta x\cos\alpha_1 + \Delta y\sin\alpha_1$$

$$N_2/K_2 = \Delta y$$

$$N_3/K_3 = \Delta x\cos\alpha_3 - \Delta y\sin\alpha_3$$

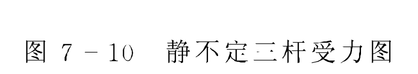
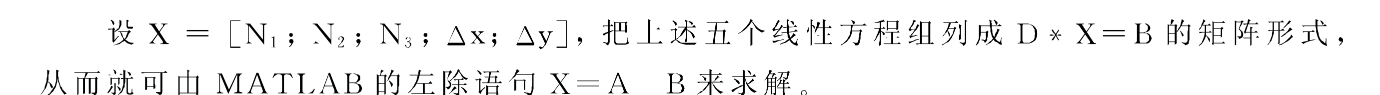


图 7－10　静不定三杆受力图



设 X ＝ [N_1；N_2；N_3；Δx；Δy]，把上述五个线性方程组列成 D * X＝B 的矩阵形式，从而就可由 MATLAB 的左除语句 X＝A　B 来求解。

◆MATLAB 程序

```
P=3000; E=200e9; L=2
A1=200e-6; A2=300e-6; A3=400e-6;
a1=pi/3; a2 = pi/2; a3=3 * pi/4;
L1=L/sin(a1); L2=L/sin(a2); L3=L/sin(a3);        % 计算杆长
K1=E * A1/L1; K2=E * A2/L2; K3=E * A3/L3;        % 计算刚度系数
D = [cos(a1), cos(a2), cos(a3), 0, 0; sin(a1),sin(a2),sin(a3),0,0; ...
     1/K1, 0, 0, -cos(a1), -sin(a1); 0, 1/K2,0,-cos(a2),-sin(a2);...
     0, 0, 1/K3, -cos(a3), -sin(a3)];        % 给系数矩阵赋值
B = [P; 0; 0; 0; 0];
format long, X = D  B        % 求解线性方程组，用长格式显示结果
```

◆程序执行结果

执行此程序，用 format long 显示的结果为

```
X = 1763.40607065591(N1)
     591.14251029634(N2)
   -2995.72429657297(N3)
       0.00016949097(dx)
       0.00001970475(dy)
```

若用普通格式显示，将得出 dy=0.0000，实际上 dy 不是零，这可从 N2 不等于零推想可知。在程序中用一个矩阵显示数值相差很大的元素时，就得采用 format long，以免丢失小的量。读者还可改变几根杆的刚度系数，看它们如何影响各杆受力的分布。

【例 7-2-2】 长为 L 的悬臂梁如图 7-11 所示，左端固定，在离固定端 L_1 处施加力 P，求它的转角和挠度。设梁 $E=200\times10^9\ N/m^2$ 和 $I=2\times10^{-5}\ m^4$ 为已知。

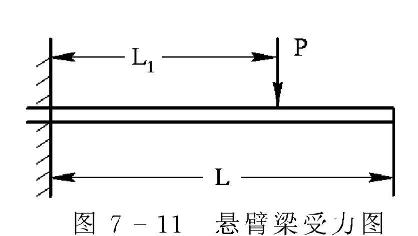


图 7-11 悬臂梁受力图

解：

◆建模

材料力学中从弯矩求转角要经过一次不定积分，而从转角求挠度又要经过一次不定积分，通常这是很麻烦而且容易出错的。而在 MATLAB 4 中，可用 cumsum 函数作近似的不定积分，只要 x 取得足够密，其结果是相当准的，且程序非常简单。在MATLAB 5 中，有更精确的函数 cumtrapz 来求不定积分。考虑到许多用 MATLAB 4 的读者，本题采用函数 cumsum。解题的关键还是在于正确地列写弯矩方程，请读者注意程序中的这部分。

本题的弯矩方程为

$$M=\begin{cases}-P(L_1-x) & (0\leqslant x\leqslant L_1)\\ 0 & (L_1\leqslant x\leqslant L)\end{cases}$$

转角 $$A=\int_0^x (M/EJ)dx$$

挠度 $$Y=\int_0^x A dx$$

◆MATLAB 程序

```
clear
L=2; P=2000; L1=1.5;            %给出已知常数
E = 200e9; I=2e-5;
x = linspace(0, L, 101); dx=L/100;     % 将 x 分 100 段，步长为 L/100
n1=L1/dx+1;                            % 确定 x=L1 处对应的下标
M1 = -P*( L1-x(1:n1));                 % 第一段弯矩赋值
M2 = zeros(1, 101-n1);                 % 第二段弯矩赋值(全为零)
M = [M1, M2];                          % 全梁的弯矩
A = cumsum(M)*dx/(E*I);                % 对弯矩积分求转角
Y = cumsum(A)*dx;                      % 对转角积分求挠度
subplot(3, 1, 1), plot(x, M), grid     % 绘弯矩图
subplot(3, 1, 2), plot(x, A), grid     % 绘弯矩图
subplot(3, 1, 3), plot(x, Y), grid     % 绘弯矩图
```

◆程序运行结果

所得的结果如图 7－12 所示。注意几根曲线之间的积分关系。本题之所以简单，是因为在 $x=0$ 处，转角和挠度都为零，因此两次积分的积分常数恰好都为零。如果它不为零，程序中就得有确定积分常数的语句，这可在下例中看到。

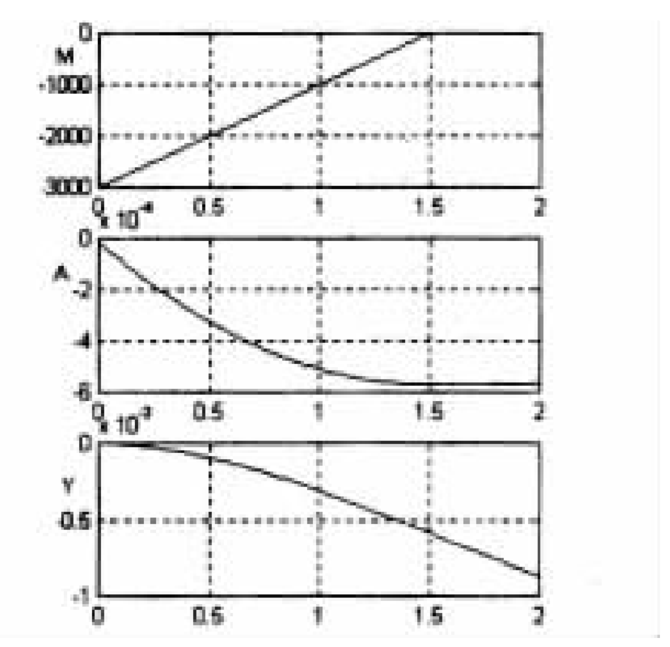


图 7－12　悬臂梁弯矩(M)，转角(A)和挠度(Y)曲线

【例 7－2－3】　简支梁受左半均匀分布载荷 q 及右边 L/4 处集中力偶 M_0 作用(如图 7－13 所示)，求其弯矩、转角和挠度。设 $L=2$ m，$q=1\ 000$ N/m，$M_0=900$ Nm，$E=200\times10^9$ N/m²，$I=2\times10^{-6}$ m⁴。

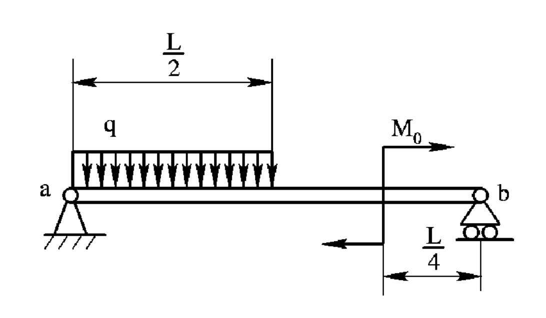


图 7－13　简支梁受力图

解：

◆建模

此题解法基本上与例 7－2－2 相同，主要差别是要处理积分常数问题。

支撑反力 N_a 和 N_b 可由平衡方程求得，设 $Q=\frac{qL}{2}$，则

$$N_a=\left(Q\cdot\frac{3}{4}L+M\right)/L,\quad N_b=Q-N_a$$

各段弯矩方程为：

$$M_1=N_ax-Q/0.5L\cdot\frac{x^2}{2}=N_a-\frac{Q}{L}\cdot x^2\quad(0\leqslant x\leqslant L/2)$$

$$M_2=N_b(L-x)+M_0\qquad(L/2\leqslant x\leqslant\frac{3}{4}L)$$

$$M_3=N_b(L-x)\qquad(\frac{3}{4}L\leqslant x\leqslant L)$$

对 M/EI 作积分，得转角 A，再作一次积分，得到挠度 Y，每次积分都要出现一个待定积分常数

$$A = \int_0^x \frac{M}{EI} \cdot dx + C_a = A_0(x) + C_a$$

此处设 $A_0(x)$=cumtrapz(M) * dx/EI。

$$Y = \int_0^x A\ dx + C_y = \int_0^x A_0(x)\ dx + C_a x + C_y = Y_0(x) + C_a x + C_y$$

此处设 $Y_0(x)$=cumtrapz(A_0) * dx。

两个待定积分常数 C_a 和 C_y 可由边界条件 Y(0)=0 及 Y(L)=0 确定：

$$Y(0) = Y_0(0) + C_y = 0$$

$$Y(L) = Y_0(L) + C_a \cdot L + C_y = 0$$

于是可得

$$\begin{bmatrix} 0 & 1 \\ L & 1 \end{bmatrix} \cdot \begin{bmatrix} C_a \\ C_y \end{bmatrix} = \begin{bmatrix} -Y_0(0) \\ -Y_0(L) \end{bmatrix}$$

即

$$\begin{bmatrix} C_a \\ C_y \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ L & 1 \end{bmatrix} \begin{bmatrix} -Y_0(0) \\ -Y_0(L) \end{bmatrix}$$

由此编成程序如下。

◆MATLAB 程序

```
%输入已知参数 L，q，M0，E，I 后，先求两铰链的支撑反力 Na 和 Nb
L=2；q=1000；M0=900；E=200e9；I=2e-6；
Na =(3 * q * L^2/8-M0)/L；Nb = (q * L^2/8+M0)/L；% 求支撑反力
x = linspace(0，L，101)；dx = L/100；
M1 = Na * x(1:51)-q * x(1:51).^2/2；    % 分三段用数组列出 M 的表达式
M2 = Nb * (L-x(52:76)) - M0；
M3 = Nb * (L-x(77:101))；M = [M1，M2，M3]；  % 列写完整的 M 数组
A0 = cumtrapz(M) * dx/(E * J)；     % 由 M 积分求转角(未计积分常数)
Y0 = cumtrapz(A0) * dx；     % 由转角积分求挠度(未计积分常数)
C=[0，1；L，1]  [-Y0(1)；-Y0(101)]；% 由边界条件求积分常数 Ca，Cy
Ca = C(1)，Cy = C(2)，
A = A0+Ca；Y=Y0+Ca * x+Cy；   % 求出转角与挠度的完整值
subplot(3，1，1)，plot(x，M)，grid   % 绘图
subplot(3，1，2)，plot(x，A)，grid
subplot(3，1，3)，plot(x，Y)，grid
```

◆程序运行结果

执行本程序的结果如图 7-14 所示。

梯形积分累加函数 cumtrapz 与定积分函数 trapz 的不同在于 cumtrapz 类似于不定积分，逐点给出积分的值，因而得出一个数列(而 trapz 只给出积分到终点的一个值)。这些函数都假定步长为 1，因此累加的值必须乘以 dx 才与积分等价。

cumtrapz 是 MATLAB 5.x 中新增的函数，可用来求面积，101 个点只能形成 100 个小面积。而 cumsum 则是把 101 个点逐次相加，相当于多算了一个点。准确地说，可以推导出

```
cumtrapz(M) =cumsum(M) - M(1)/2 -M/2
```

实际上只要点取得足够多，在 MATLAB 4.x 中，直接用 cumsum(M)代替 cumtrapz(M)也是可以接受的。

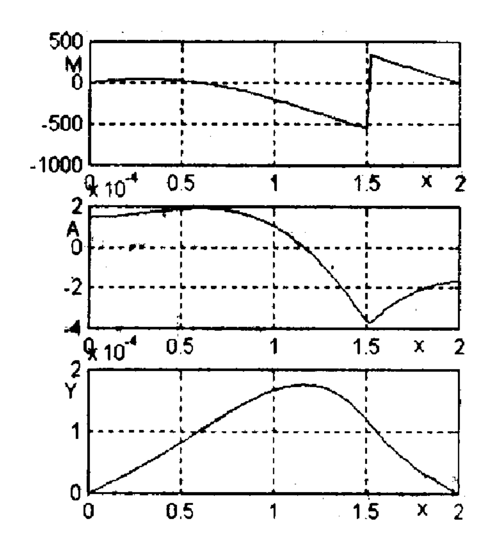


图 7－14　例 7－2－3 的弯矩、转角和挠度曲线

【例 7－2－4】　材料力学复杂应力状态的分析——Moore 圆。

解：

◆建模

已知两正交截面上的正应力 σ_x、σ_y 和剪应力 τ_{xy}，求其它斜截面上的应力值，如图 7－15 所示。

根据应力状态理论的结果，求任意斜截面上正应力 σ 和剪应力 τ 的公式为：

$$\sigma=\frac{\sigma_x+\sigma_y}{2}+\frac{\sigma_x-\sigma_y}{2}\cos2\alpha-\tau_{xy}\sin2\alpha$$

$$\tau=\frac{\sigma_x-\sigma_y}{2}\sin2\alpha+\tau_{xy}\cos2\alpha$$

其轨迹可表示为一个圆，其正负号规则如下：

σ——拉应力为正，压应力为负

τ——对立方体中心顺时针的剪应力为正

α——反时针为正

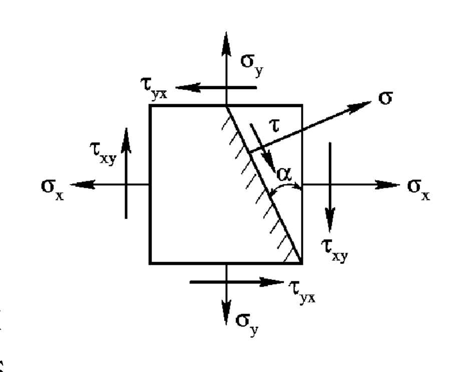


图 7－15　复杂应力状态的分析

由此可编成 MATLAB 程序来计算并绘制其应力圆，因 MATLAB 不接受希腊字符，程序中取 S＝σ，T＝τ，a＝α。

◆MATLAB 程序

```
% 输入两方向的正应力和剪应力，S=σ，T=τ
Sx=input('Sx(MPa) = ');
Sy=input('Sy(MPa) = '); Txy=input('Txy(MPa) = ');
a=linspace(0, pi, 37);     % 将应力圆上的圆周角分为 36 份
Sa=(Sx+Sy)/2; Sd=(Sx-Sy)/2;
sigma=Sa+Sd*cos(2*a)-Txy*sin(2*a);     % 应力圆方程
tau = Sd*sin(2*a)+Txy*cos(2*a);
plot(sigma, tau, Sx, Txy, '*');     % 绘图并绘出基准点
axis equal;             % 使 x 和 y 轴取等比例
v=axis;                 % 找坐标轴的边界 v=[xmin, xmax, ymin, ymax]
line([v(1), v(2)], [0, 0]);     % 画出 x, y 坐标
line([0, 0], [v(3), v(4)])
hold, plot(Sa, 0, 'x')       % 标出应力圆的圆心
Smax=max(sigma), Smin=min(sigma), Tmax=max(tau)
% 以下是求任意斜截面 a 角上 σ 及 τ 的程序段
```

```
h=input('若不求应力，键入 0。若要再求应力，键入 1 ')；
while h~=0；
a=input('给出斜截面方向角 a=(弧度)') % 给出斜截面方向，求其应力状态
sigma=Sa+Sd * cos(2 * a)-Txy * sin(2 * a)
tau= Sd * sin(2 * a)+Txy * cos(2 * a)
plot(sigma, tau, 'or')        % 画出待求的应力状态点
h=input('若不继续求应力，键入 0。若还要求应力，键入 1')；
end, hold off
```

◆程序运行结果

运行此程序，若输入为(单位 MPa)：

Sx=20

Sy=0

Txy=5

得出

Smax = 21.1603

Smin = -1.1603

Tmax = 11.1603

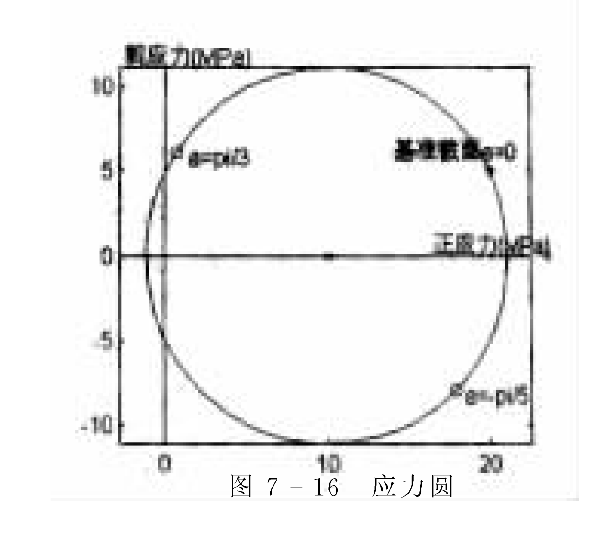


图 7-16 应力圆

并得出 Moore 圆如图 7-16 所示。

给定 a=pi/3 及 a=-pi/5 求得图中相应的两点应力为：

a=pi/3 处 sigma = 0.6699,

tau = 6.1603

a=-pi/5 处 sigma =17.8455,

tau = -7.9655

【例 7-2-5】 拉弯合成部件的截面设计。这一设计计算将归结为解一个三次代数方程，过去要用试凑法反复运算，本例显示了用 MATLAB 求解的简洁。钻床立柱如图 7-17 所示。设P=15 kN，许用拉应力[σ]=35 MPa，钻头轴与立柱轴距离为 0.4 m，试求立柱直径。

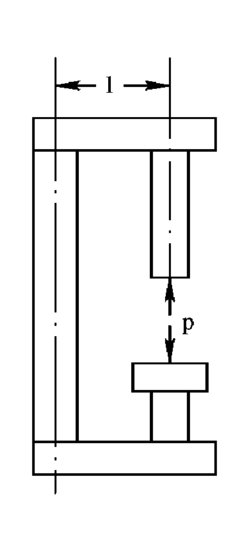


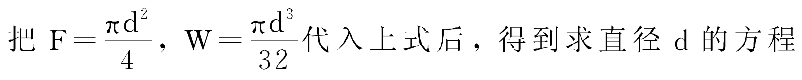
图 7-17 钻床受力图

解：

◆建模

立柱受到拉力 P 和弯矩 Pl，两者产生的拉应力之和为最大拉应力，令它小于[σ]

$$\sigma=\frac{P}{F}+\frac{Pl}{W}\leqslant[\sigma]$$



把 $F=\frac{\pi d^2}{4}$，$W=\frac{\pi d^3}{32}$代入上式后，得到求直径 d 的方程

$$\frac{[\sigma]\cdot\pi}{32}d^3-\frac{P}{8}d-Pl\geqslant 0$$

这个三次代数方程可用 MATLAB 多项式求根的 roots 函数求解。

◆MATLAB 程序

```
P=input('P='), l=input('l='),              % 输入力和偏心距
asigma=input('[σ]='),                      % 输入许用拉应力 asigma
a=[asigma*pi/32, 0, -P/8, -P*l];           % 求三次代数方程的系数向量
r=roots(a);                                % 求代数方程的根
d=r(find(imag(r)==0))                      % 只取实根
```

◆程序运行结果

运行此程序，按提示输入以下条件

P = 15000，l =0.4，[σ] = 35e6

得到的解为

d = 0.1219 m

7.3 机 械 振 动

【例 7-3-1】 分析单自由度阻尼系统的阻尼系数对其固有振动模态的影响。

解：

◆建模

单自由度阻尼系统振动方程如下

$$m\ddot{x}+c\dot{x}+kx=0$$

化成

$$\ddot{x}+\zeta\omega\dot{x}+\omega_n^2x=0$$

其中，$\omega_n=\sqrt{\frac{k}{m}}$，$\zeta=\frac{c}{2}\sqrt{\frac{1}{mk}}=\frac{c}{\sqrt{2mk}}$，其解为

$$x(t)=Ae^{-\zeta\omega_n t}\sin(\omega_d t+\varphi)$$

其中，$A=\sqrt{\frac{(v_0+\zeta\omega_0x_0)^2+(x_0\omega_d)^2}{\omega_d^2}}$，$\varphi=tg^{-1}\left(\frac{x_0\omega_d}{v_0+\zeta\omega_nx_0}\right)$，$\omega_d=\omega_n\sqrt{1-\zeta^2}$。

x_0——初始位置

v_0——初始速度

分别设 ζ =0.1 到 1，公共参数为 $\omega_n=10$；$x_0=1$；$v_0=0$；计算的终点时间 $t_f=2$。现用 MATLAB 编程求出其解并画出波形。

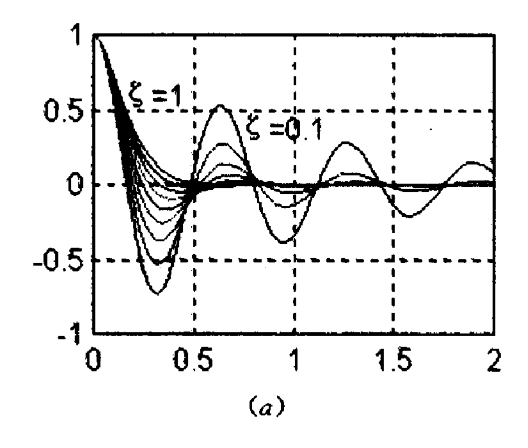
◆MATLAB 程序

```
clear, wn=10; tf=2; x0=1; v0=0;
for j=1:10
   zeta(j)=0.1*j;            % 设定不同的 ζ
   wd(j)=wn*sqrt(1-zeta(j)^2);        % 求 ωd
   a=sqrt((wn*x0*zeta(j)+v0)^2+(x0*wd(j))^2)/wd(j); % 求振幅 A
   phi=atan2(wd(j)*x0,v0+zeta(j)*wn*x0); % 用 atan2 是为了求四象限相角
   t=0: tf/1000: tf;            %设定自变量数组
   x(j, :)=a*exp(-zata(j)*wn*t).*sin(wd(j)*t+phi);       %求过渡过程
end
```

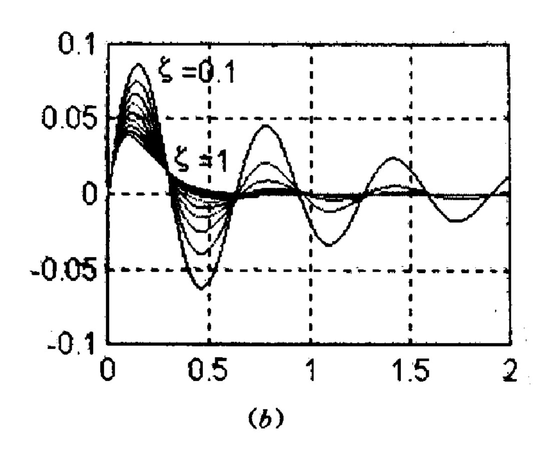
```
plot(t, x(1,:), t, x(2,:), t, x(3,:), t, x(4,:), t, x(5,:), ...   % 绘图
t, x(6,:), t, x(7,:), t, x(8,:), t, x(9,:), t, x(10,:))
grid, figure, mesh(x)      % 画出三维图形
```

◆程序运行结果

执行此程序即可得到图 7－18(*a*)所示结果。改变初始条件为 $x_0=0$，$v_0=1$ 可得到图 7－18(*b*)所示结果。实际上后一组曲线就是系统的脉冲过渡函数。因为脉冲函数的幅度是无穷大，而持续时间却是无限小，其面积为一个单位。因此脉冲激励的最后效果(在 t=+eps 处)可形成一个单位的初速 v_0，由它产生的波形就是脉冲过渡函数。



(*a*)



(*b*)

图 7－18 不同 ζ 对固有振动模态的影响((*a*)中 $x_0=1$，(*b*)中 $v_0=10$)

这个方法的缺点是整个程序都以手工推导的公式为基础，这既费事又容易出错，最好从原始方程出发，不经手工推导而靠 MATLAB 直接求解。下一个例子中我们将采用 4.3 节中的极点留数求解法以简化程序。

画出的三维图形如图 7－19 所示，从中可以更形象地看出 ζ 对固有振动模态的影响，因为没有彩色，所以视觉效果要差一些，读者在计算机屏幕上看要好得多，并且可以再键入 rotate3d 命令，以使用鼠标拖动三维图形旋转，获得更清晰的概念。

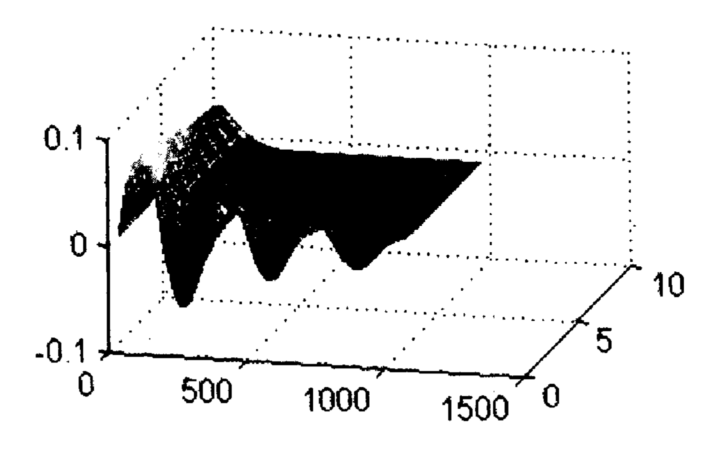


图 7－19 不同 ζ 对固有振动模态影响的三维图

【例 7－3－2】 设单自由度阻尼系统的质量 m=1 kg，弹簧刚度系数 K=100 N/m，速度阻尼系数 c=4 N·s/m，求它在如下外力作用下的强迫振动，得出 t≤1.2 s 的波形。

$$f=\begin{cases} t/0.015 & (0\leqslant t\leqslant 0.15\ \text{s}) \\ 10 & (0.15<t\leqslant 1.2\ \text{s}) \end{cases}$$

解：

◆建模

首先求出系统的脉冲过渡函数 h(t)，则强迫振动的波形就等于 h(t)和外加力 f(t)作卷积的结果

$$x(t) = \int_0^\tau h(t-\tau)\cdot f(\tau)d\tau$$

在 MATLAB 中，h(t)和 f(t)都可用数组 h 和 f 表示，其取样间隔 dt 应相同，便有

$$x=conv(h, f) * dt$$

卷积计算很繁，通常讲振动时只讨论一些标准的有解析表示式的激励信号，如方波、正弦波等，而用了 MATLAB 就不必受限制。在本题中脉冲过渡函数用极点留数函数 residue 求得，然后用卷积函数 conv 根据输入函数和脉冲过渡函数求输出。

◆MATLAB 程序

```
m=1; c=4; K=100; dt=0.015;           % 输入给定的参数
w0=sqrt(K/m);                        % 求系统固有频率
zeta=c/sqrt(m*K)/2;                  % 求系统固有阻尼系数
a=[1, 2*zeta*w0, w0^2]; b=1;         % 求分母、分子的系数
[r, p]=residue(b, a);                % 求极点、留数
t=0:dt:1.2;
h=r(1)*exp(p(1)*t)+r(2)*exp(p(2)*t);     % 求出系统的脉冲响应
f=[1:10, 10*ones(1, 70)];                % 给出外加力的采样值
x=conv(h, f)*dt;          % 把脉冲响应和外加力作卷积
plot(t(1:80), x(1:80))                   % 绘图
v1=diff(x)/dt;       % 求导得出速度，注意求导后数组长度少 1
[t(1:80)', f(1:80)', x(1:80)', [0, v1(1:79)]']     % 列出结果
```

◆程序运行结果

执行此程序的结果见图 7－20 所示。其大量的数值结果予以删略。读者不妨把程序中的 h 改用例 7－3－1 的方法在 $v_0=1$ 条件下求出，其所得结果应该完全相同。

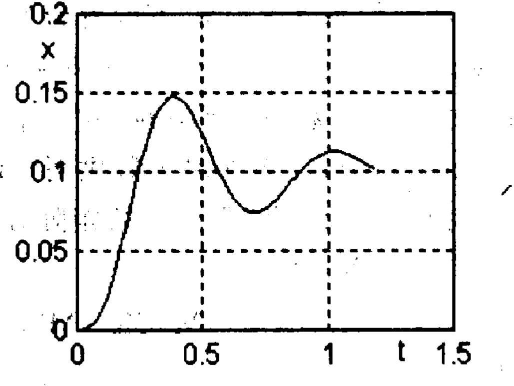


图 7－20　单自由度阻尼系统的强迫振动

【例 7－3－3】　二自由度可解耦系统的振动模态分析。

（注：本例在第一版中以传统的解法为主，很繁且不能体现利用 MATLAB 的优越性，所以在第二版中删去了这种方法，改为直接用矩阵计算的方法，其程序为 ex733a。考虑到第一版读者的需要，在下载程序集中仍保留了老方法的程序 ex733。）

图 7－21 表示了一个由两个质量和两个弹簧及阻尼器构成的二自由度振动系统，今要在给定两个质量的初始位置和初始速度的情况下求系统的运动。

解：

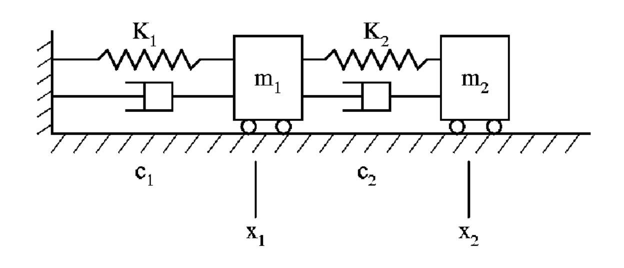


图 7－21 二自由度振动模型

◆建模

设 x_1 和 x_2 分别表示两个质量关于它们的平衡位置的偏差值，则此二自由度振动系统的一般方法为

$$\left.\begin{aligned} m_1\ddot{x}_1+(c_1+c_2)\dot{x}_1-c_2\dot{x}_2+(k_1+k_2)x_1-k_2x_2=0 \\ m_2\ddot{x}_2+c_2(\dot{x}_2-\dot{x}_1)+k_2(x_2-x_1)=0 \end{aligned}\right\} \tag{1}$$

可写成矩阵形式

$$M\ddot{X}+C\dot{X}+KX=0 \tag{2}$$

其中

$$M=\begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix},\ C=\begin{bmatrix} c_1+c_2 & -c_2 \\ -c_2 & c_2 \end{bmatrix},\ K=\begin{bmatrix} k_1+k_2 & -k_2 \\ -k_2 & k_2 \end{bmatrix},\ X=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3}$$

这是一个四阶常微分方程组。给出它的初始条件(初始位置 X_0 和初始速度 X_{d0})：

$$X_0=\begin{bmatrix} X_{10} \\ X_{20} \end{bmatrix},\ X_{d0}=\dot{X}_0=\begin{bmatrix} \dot{x}_{10} \\ \dot{x}_{20} \end{bmatrix}=\begin{bmatrix} x_{d10} \\ x_{d20} \end{bmatrix} \tag{4}$$

可以求出它的解。但用解析方法求解非常麻烦，如果 $C=0$，即无阻尼情况时，则系统可解耦为两种独立的振动模态，通常书上只给出作解耦简化后的解。

有了 MATLAB 工具，根本无需设 $C=0$，也无需解耦，就可以用很简单的程序求出其数值解。其基本思路是把原始方程化成典型的四个一阶方程构成的状态方程组

$$\dot{Y}=AY \tag{5}$$

此方程在初始条件 $Y=Y_0$ 下的解为 $Y=Y_0e^{At}$。用 MATLAB 表示为 Y=Y0 * expm(A * t)。其中 expm 表示把(A * t)看成矩阵来求其指数。在例 5－3－2 中给出的是标量 x 求指数的方法，求矩阵指数当然要麻烦一些。但概念是相仿的，我们不必都弄清细节。MATLAB 提供了 expm，expm1 等多种函数供用户调用。所以，我们只要把 Y，Y0 和 A 找到就行。

先把方程(2)写成两个一阶矩阵方程

$$\left.\begin{aligned} \dot{X}&=X_d \\ \dot{X}_d&=\ddot{X}=-M\quad C\dot{X}-M\quad KX \end{aligned}\right\}\rightarrow\begin{bmatrix} \dot{X} \\ \dot{X}_d \end{bmatrix}=\begin{bmatrix} 0 & I \\ -M\quad K & -M\quad C \end{bmatrix}\begin{bmatrix} X \\ X_d \end{bmatrix} \tag{6}$$

于是

$$Y=\begin{bmatrix} X \\ X_d \end{bmatrix},\ Y_0=\begin{bmatrix} X_0 \\ X_{d0} \end{bmatrix},\ A=\begin{bmatrix} 0 & I \\ -M\quad K & -M\quad C \end{bmatrix} \tag{7}$$

对于本题的二自由度系统，$X=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$和 $X_d=\begin{bmatrix} x_{d1} \\ x_{d2} \end{bmatrix}$，所以 Y 和 Y_0 都是 4×1 的单列数

组；由于 A 中的四个元素都是 2×2 方阵，因此 A 是 4×4 方阵。对于更多自由度的系统，公式(7)都是正确的。只要改变 O、I、M、K、C、Y、Y_0 的阶数即可。

下面给出二自由度系统的一个数值例，设 m1=1；m2=9；k1=4；k2=2；cl 和 c2 可由用户输入。求在初始条件 x0=[1；0]和 xd0=[0；−1]下，系统的输出 x1，x2 曲线。

◆MATLAB 程序

根据上面的模型可以写出程序 cx733a 如下。

```
m1=1; m2=9; k1=4; k2=2;          % 输入各原始参数
c1=input('c1='); c2=input('c2=');        % 输入阻尼系数
x0=[1; 0]; xd0=[0; -1]; tf=50; dt=0.1;       % 给出初始条件及时间向量
M=[m1,0; 0,m2]; K=[k1+k2, -k2; -k2, k2];      % 构成二阶参数矩阵
C=[c1+c2, -c2; -c2, c2];
A=[zeros(2, 2), eye(2); -M   K, -M   C];      % 构成四阶参数矩阵
y0=[x0; xd0];          % 四元变量的初始条件
for i=1:round(tf/dt)+1          % 设定计算点，作循环计算
    t(i)=dt * (i-1);
    y(:,i)=expm(A * t(i)) * y0;          % 循环计算矩阵指数
end
subplot(2,1,1), plot(t, y(1,:)), grid      % 按两个分图绘制 x1，x2 曲线
subplot(2,1,2), plot(t, y(2, :)), grid
```

◆程序运行结果

运行此程序，输入 c1=0.2，c2=0.5 所得的结果如图 7-22 所示。从中可清楚地看到振动的两种模态。特别是 x1 的运动反映了两种模态的叠合。给出不同的初始条件，各模态的幅度也会变化。输入 c1=0，c2=0 所得的结果如图 7-23 所示，这时两种振动模态是档解耦的。可以看出，用计算机解题时，问题和数据的复杂性对解题的过程毫无影响。通常我们选择比较简单，且有解析解的数据作为检验程序之用。一旦确定程序正确，它就可用在很复杂的情况下。例如，作者就曾把 ex733a 的核心语句用在一个五自由度的系统上，解一个 10 阶的线性方程组，同样可得出满意的结果。

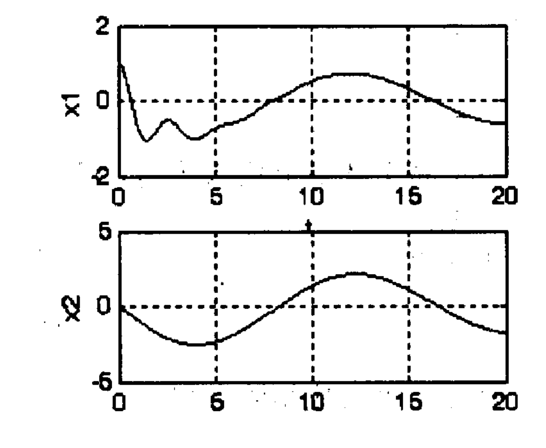


图 7-22　二自由度振动的输出波形

(c1=0.2，c2=0.5)

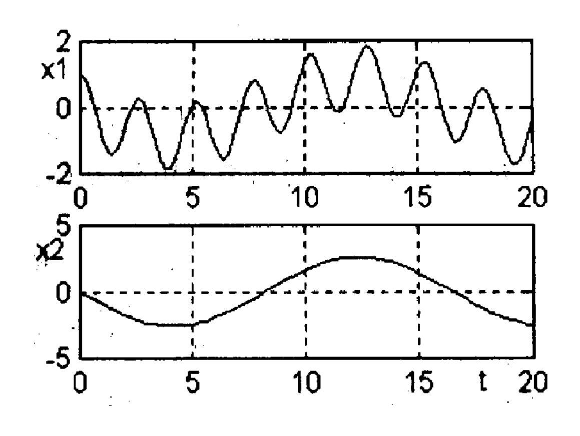


图 7-23　二自由度振动的输出波形

(c1=c2=0)

第 8 章　在电工和电子线路中的应用举例

8.1　在电工原理中的应用

【例 8－1－1】 按图 8－1 所示的梯形直流电路，问

(1) 如 $U_S=10$ V，求 U_{bc}，I_7，U_{dc}；

(2) 如 $U_{dc}=4$ V，求 U_{bc}，I_7，U_S。

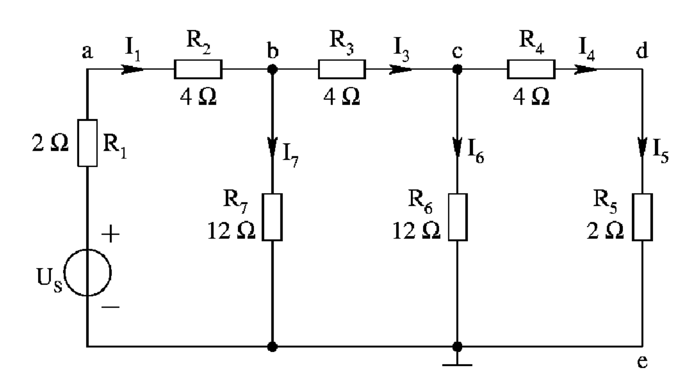


图 8－1　例 8－1－1 的电路图

解：

◆建模

此电路中设节点电压为变量，共有 U_a，U_b，U_c，U_d等四个。

各支路电流均用这些电压来表示

$$I_1=\frac{U_S-U_b}{R_1+R_2}\qquad I_3=\frac{U_b-U_c}{R_3}\qquad I_7=\frac{U_b}{R_7}$$

$$I_4=I_5=\frac{U_c}{R_4+R_5}\qquad I_6=\frac{U_c}{R_6}$$

对 b 点和 c 点列出节点电流方程：$I_1=I_3+I_7$，$I_3=I_4+I_6$，将上述各值代入化简，得

$$\left(\frac{1}{R_3}+\frac{1}{R_1+R_2}+\frac{1}{R_7}\right)U_b-\frac{1}{R_3}U_c=\frac{U_S}{R_1+R_2}\tag{1}$$

$$\frac{U_b}{R_3}-\left(\frac{1}{R_3}+\frac{1}{R_4+R_5}+\frac{1}{R_6}\right)U_c=0\tag{2}$$

当问题(1)给出 U_S时，这两个方程中只有两个未知数 U_b 和 U_c，可写成矩阵方程

$$A_1\cdot\begin{bmatrix}U_b\\U_c\end{bmatrix}=\begin{bmatrix}a_{11}&a_{12}\\a_{21}&a_{22}\end{bmatrix}\begin{bmatrix}U_b\\U_c\end{bmatrix}=\begin{bmatrix}\dfrac{U_S}{R_1+R_2}\\0\end{bmatrix}=\begin{bmatrix}a_{13}\\a_{23}\end{bmatrix}U_S$$

读者可从方程(1)，(2)中找到 a_{11}，a_{12}，a_{21}，a_{22} 及 a_{13}，a_{23} 的表达式，并用矩阵左除解出 U_b，U_c。

由 $U_{bc}=U_b-U_c$，$U_{dc}=\dfrac{R_5}{R_4+R_5}U_c$，$I_7=\dfrac{U_b}{R_7}$，即可得到解问题(1)。

对于问题(2)，可以推出类似的矩阵表达式，只是输入输出量不同，请读者自行推导，并与下面的 MATLAB 程序对照。由于同一系统的系数矩阵有许多相同的项，编程时可以利用这个特点来简化其赋值过程。

◆MATLAB 程序

直流电路

```
r1=2; r2=4; r3=4; r4=4; r5=2; r6=12; r7=12;     %为元件赋值
a11=1/(r1+r2)+1/r3+1/r7; a12=-1/r3; a13=1/(r1+r2)    %将各系数赋值
a21=1/r3; a22=-1/r3-1/(r4+r5)-1/r6; a23=0
us=input('us=');                % 输入问题(1)的已知条件
A1=[a11, a12; a21, a22]         % 列出系数矩阵 A1
u=A1\[a13*us; 0]                % u=[ub; uc]
ubc=u(1)-u(2), ude=u(2)*r5/(r4+r5), i=u(1)/r7    % 解问题(1)
ude=input('ude='),            % 输入问题(2)的已知条件
A2=[A1, -[a13; a23]; 0, r5/(r4+r5), 0]           % 列出系数矩阵 A2
u=A2\[0; 0; ude],                                % u=[ub; uc; us]
ubc=u(1)-u(2),i=u(1)/r7,us=(r1+r2)*(u(1)*a11+u(2)*a12)
                                                 % 解问题(2)
```

◆程序运行结果

(1) 输入 us=10 时，ubc = 2.2222，udc = 0.7407，i7 = 0.3704；

(2) 输入 ude=4 时，ubc = 12.0000，i7 = 2.0000，us = 54.0000。

【例 8-1-2】 如 8-2 图所示电路，在 t<0 时，开关 S 位于“1”，电路已处于稳态，t=0 时，开关 S 闭合到“2”，求 U_C 和 I_{R2} 的响应，并画出它们的波形。

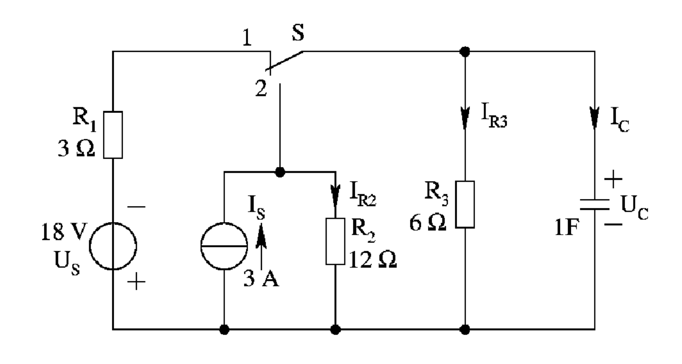


图 8-2　例 8-1-2 的电路图

解：

◆建模

这是一个分析暂态过程的问题。先要找到其初值和终值。

在 $t=0_-$ 时　$U_C(0_-)=-12$ V，$I_C(0_-)=0$；当 $t=0_-$ 时，因为电容器端电压不可能突

变，仍有 $U_C(0_-)=U_C(0_-)=-12$ V，电流源向两个电阻和一个电容的并联系统供电，两个电阻的电流应等于电容电压除以电阻，即

$$I_{R2}(0_-)=\frac{U_C(0_-)}{R_2}=-1\ \text{A}$$

$$I_{R3}(0_-)=\frac{U_C(0_-)}{R_3}=-2\ \text{A}$$

电容的充电电流为电流源总电流减去电阻电流，故

$$I_C(0_-)=I_S-I_{R2}(0_+)-I_{R3}(0_+)=3-(-1)-(-2)=6\ \text{A}$$

再分析终值，达到稳态后，电容中将无电流，电流源的全部电流将在两个电阻之间分配，其端电压应相同，它也就是电容上的终电压，结果应为 $U_{Cf}=12$ V，$I_{R2f}=1$ A。

初值和终值之间的过渡波形按三要素法计算。

◆MATLAB 程序

```
r1=3; us=18; is=3; r2=12; r3=6; C=1;        % 给出原始数据
uc0=-12; ir20=uc0/r2; ir30=uc0/r3;          % 算出初值
ic0=is-ir20-ir30;
ir2f=is*r3/(r2+r3);                         % 算出终值
ir3f=is*r2/(r2+r3);
ucf=ir2f*r2; icf=0;
t=[-2:0]-eps, 0:15];    % 注意时间数组的设置，在 t=0 附近设两个点
uc(1: 3)=-12; ir2(1:3)=3;                   % t<0 时的值
T = r2*r3/(r2+r3)*C;                        % 求充电时常数
uc(4: 19)=ucf+(uc0-ucf)*exp(-t(4:19)/T);
ir2(4: 19)=ir2f+(ir20-ir2f)*exp(-t(4:19)/T);    % 用三要素法求输出
subplot(2, 1, 1); h1=plot(t, uc),       % 绘电压 uc 波形
grid, set(h1, 'linewidth', 2)           % 加大线宽
subplot(2, 1, 2), h2=plot(t, ir2);      % 绘电流 ir2 波形
grid, set(h2, 'linewidth', 2)
gtext('uc'), gtext('ir2')
```

◆程序运行结果

执行此程序的结果如图 8－3 所示。

【例 8－1－3】 如图 8－4 所示电路，已知 $R=5\ \Omega$，$\omega L=3\ \Omega$，$\frac{1}{\omega C}=5\ \Omega$，$\dot{U}_C=10\angle 0°$，求 $\dot{I}_R$，$\dot{I}_C$，$\dot{I}$ 和 $\dot{U}_L$，$\dot{U}_S$，并画其相量图。

解：

◆建模

这是普通的交流稳态电路问题，其方程如下：

设 $Z_1=j\omega L$，$Z_2=R$，$Z_3=1/j\omega C$

R 与 C 并联后的阻抗为

$$Z_{23}=\frac{Z_3\cdot Z_2}{Z_2+Z_3}$$

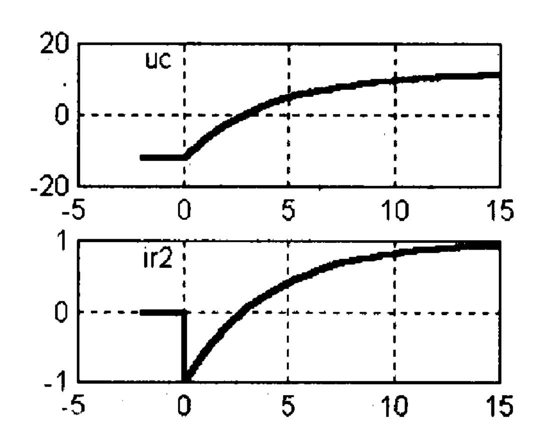


图 8－3　U_C 和 I_{R2} 的波形

总阻抗为　　$Z=Z_1+Z_{23}$

$\dot{I}=\dot{U}_S/Z$　　$\dot{U}_L=\dot{I}\cdot Z_1$　　$\dot{U}_C=\dot{I}\cdot Z_{23}$

$\dot{I}_R$ 及 $\dot{I}_C$ 可由 $\dot{U}_C$ 分别除以 Z_2 及 Z_3 得到

$\dot{I}_R=\dot{U}_C/Z_2$　　$\dot{I}_C=\dot{U}_C/Z_3$

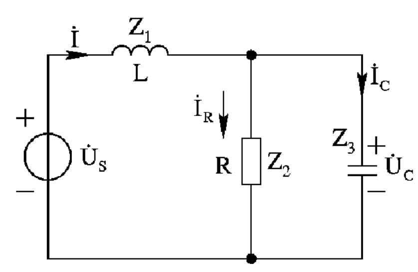


图 8－4　例 8－1－3 电路图

在 MATLAB 中，任何一个变量的元素，都可以是复数，它可以代表电压和电流相量，也可以表示复数阻抗，无需特别注明，所以程序中没有(也不允许有)字母上的“点”号，以后不再向读者说明。

◆MATLAB 程序(注意它的复数运算)

```
z1＝3 * j; z2＝5; z3＝5/j; uc＝10;
z23＝z2 * z3/(z2＋z3); z＝z1＋z23;
Ic＝uc/z3, Ir＝uc/z2, I＝Ic＋Ir, uL＝I * z1, us＝I * z
disp('    Ir      Ic      I      ul      us ')
disp('幅值'), disp(abs([Ir, Ic, I, uL, us]))
disp('相角'), disp(angle([Ir, Ic, I, uL, us]) * 180/pi)
% compass 是 MATLAB 中绘制复数相量图的命令，用它画相量图特别方便
ha＝compass([Ir,Ic,I,uL,us]); % ha 是本图的图柄，如不需改变线宽，可省去它
set(ha, 'linewidth', 2);     % 把向量线条加粗至 2 mm
```

◆程序运行结果

	Ir	Ic	I	uL	us
幅值	2.0000	2.0000	2.8284	8.4853	7.2111
相角	0	90.0000	45.0000	135.0000	56.3099

画出的相量图如图 8－5 所示。我们可以再多次用 gtext('Ir')等命令给图上各向量加上标注。这里显示的是在 MATLAB 4.x 中 compass 命令生成的图形，其坐标是直角坐标，用 MATLAB 5.x 时，compass 命令显示的是极坐标，在例 8－1－5 中将给出它的图形，两者的其他用法没有什么差别。

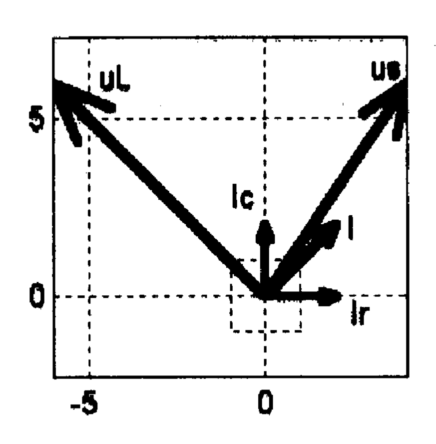


图 8－5 例 8－1－3 所得的相量图

【例 8－1－4】 如图 8－6 所示电路，已知 $\dot{U}_S=10+10\cos t$，$\dot{I}_S(t)=5+5\cos 2t$，求 $\dot{U}(t)$。

解：

◆建模

这是一个含三个频率分量的稳态交流电路问题，可以按每个频率成份分别计算，再叠加起来。但是，更高明的办法是利用 MATLAB 的元素群计算特性，把多个频率分量及相应的电压、电流、阻抗等都看做多元素的行数组，每一元素对应于一种频率分量的值。因为它们服从同样的方程，所以程序就特别简洁。

(1) 先看 $\dot{U}_S$ 对 b、d 点产生的等效电压 $\dot{U}_{OC}$，假定电流源开路，电桥电路可得

$$\dot{U}_{OC}=\left[\frac{Z_2}{Z_1+Z_2}-\frac{Z_4}{Z_3+Z_4}\right]\cdot\dot{U}_S$$

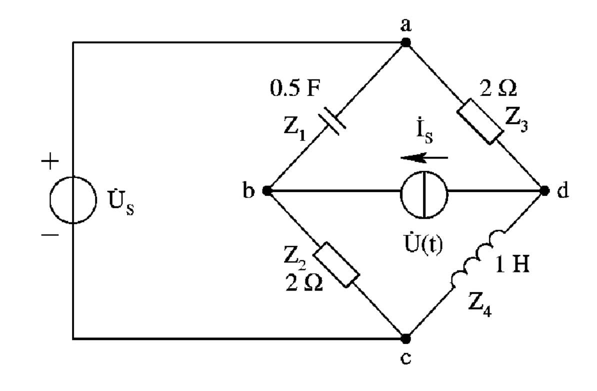


图 8－6 例 8－1－4 的电路图

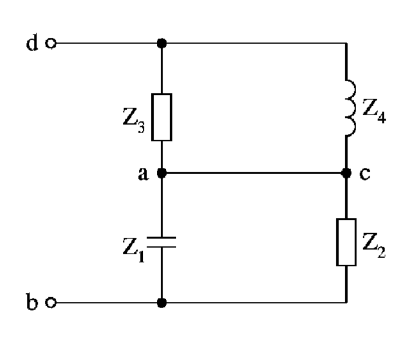


图 8－7 求等效内阻的图

(2) 根据戴维南定理，U_S 的等效电流源的内阻应计算如下：设 U_S 短路，求由 b，d 向网络看的阻抗，其等效电路如图 8－7 所示。

$$Z_{eq}=\frac{Z_3Z_4}{Z_3+Z_4}+\frac{Z_1Z_2}{Z_1+Z_2}$$

电流源在 b，d 间产生的电压为 $I_S\cdot Z_{eq}$。

(3) 根据叠加原理

$$\dot{U}=\dot{I}_S\cdot Z_{eq}+\dot{U}_{OC}$$

◆MATLAB 程序

```
clear, format compact
```

```
w=[eps, 1, 2];us=[10, 10, 0];Is=[5, 0, 5]; % 按三种频率设定输入信号数组
z1=1./(0.5*w*j);z4=1*w*j;      % 电抗分量是频率的函数，故自动成为数组
z2=[2, 2, 2]; z3=[2, 2, 2];     % 对电阻分量也列写成常数数组
uoc=(z2./(z1+z2)-z4./(z3+z4)).*us;   % 列出电路的复数方程
zeq=z3.*z4./(z3+z4)+z1.*z2./(z1+z2);   % 列出等效阻抗
u=Is.*zeq+uoc;                  % 求解
disp('    w     um     phi ')     % 显示
disp([w', abs(u'), angle(u')*180/pi])
```

◆程序运行结果

```
     w          um          phi
  0.0000    10.0000          0
  1.0000     3.1623   -18.4349
  2.0000     7.0711    -8.1301
```

由此我们可以写出 u 的表达式为

$$u=10+10\cos(t+18.434\,9°)+7.071\,1\cos(2t+8.130\,1°)$$

思考题：

(1) 对直流分量，我们不用零作为其频率而用 eps(相对精度)，是什么原因？

(2) 如果输入电压为 us=10+10 sin(t)，该程序中应如何改变？

(3) 注意比较本程序中最后两个 disp 语句的不同。

【例 8-1-5】　如图 8-8 所示电路，设

$$R_1=2\ \Omega,\ R_2=3\ \Omega,\ R_3=4\ \Omega$$

$$jX_L=j2,\ -jX_{C1}=-j3,\ -jX_{C2}=-j5,$$

$$\dot{U}_{S1}=8\angle 0\ \text{V},\ \dot{U}_{S2}=6\angle 0\ \text{V},\ \dot{U}_{S3}=8\angle 0\ \text{V},\ \dot{U}_{S4}=15\angle 0\ \text{V},$$

求各支路的电流相量和电压相量。

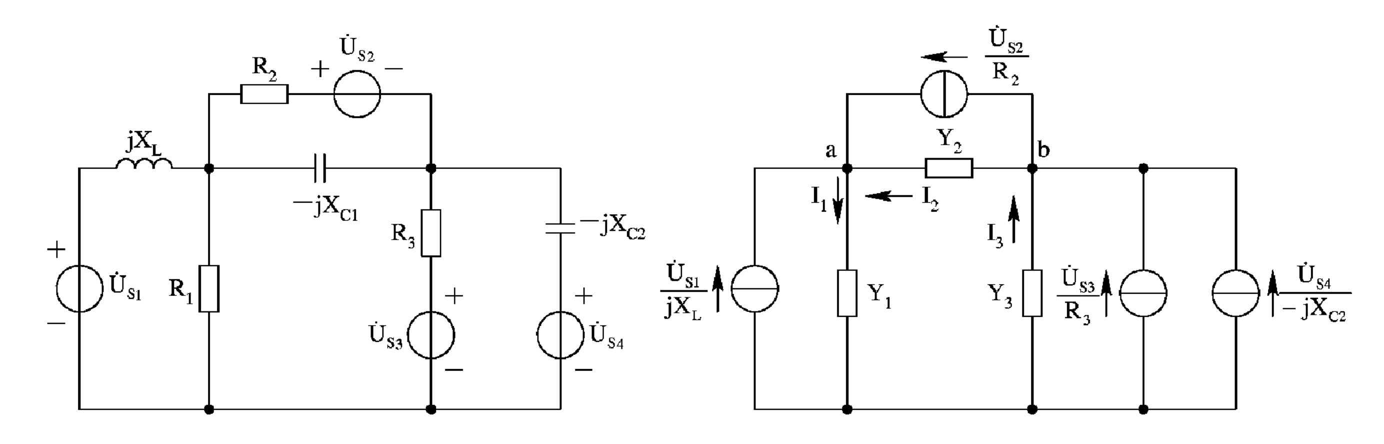


图 8-8　例 8-1-5 的电路图

解：

◆建模

先把原图中的电压源换成等效的电流源，则导纳

$$Y_1=\frac{1}{R_1}+\frac{1}{jX_L},\ Y_2=\frac{1}{R_2}-\frac{1}{jX_{C1}},\ Y_3=\frac{1}{R_3}-\frac{1}{jX_{C2}}$$

均为两并联元件导纳之和，按图示电源方向，其电流为

$$\dot{I}_1=\dot{U}_aY_1,\ I_2=(\dot{U}_b-\dot{U}_a)Y_2,\ I_3=-\dot{U}_bY_3$$

列出 a，b 两点的电流方程

$$Y_1\dot{U}_a-Y_2(\dot{U}_b-\dot{U}_a)=\dot{U}_{S1}/jX_L+U_{S2}/R_1$$

$$Y_2(\dot{U}_b-\dot{U}_a)+Y_3\dot{U}_b=\dot{U}_{S3}/R_3\cdot\dot{U}_{S1}/jX_{C2}-\dot{U}_{S2}/R_2$$

从这两个联立方程可写成 $A\cdot\begin{bmatrix}U_a\\U_b\end{bmatrix}=B$ 的矩阵形式。

◆MATLAB 程序

```
R1=2; R2=3; R3=4; XL=2; XC1=3; XC2=5;      % 给出原始数据
us1=8, us2=6; us3=8; us4=15;               % 给出原始数据
Y1=1/R1+1/(j*XL);                          % 用复数表示各支路导纳
Y2=1/R2-1/(j*XC1);
Y3=1/R3-1/(j*XC2);
A=[Y1+Y2, -Y2; -Y2, Y2+Y3];    % 按线性方程组列出 ua, ub 的系数矩阵
% 列出线性方程组右端
B=[us1/(j*XL)+us2/R1; us3/R3+us4/(-j*XC2)-us2/R2];
U=A  B; ua=U(1), ub=U(2)                   % 求 ua, ub
I1=ua*Y1, I2=(ub-ua)*Y2, I3=ub*Y3,         % 求各支路的 I
I1R=ua/R1, I1L=ua/(j*XL),
I2R=(ub-ua)/R2, I2C=(ub-ua)/(-j*XC1),
I3R=ub/R3, I3C=ub/(-j*XC2),
H=compass([ua, ub, I1, I2, I3]);      % 画相量图，设定此图的图柄为 H
set(H, 'linewidth', 2)                % 改变相量图线宽
```

◆程序运行结果

```
ua = 4.8845 - 0.5981i
ub = 5.4874 + 2.5752i
I1 = 2.1432 - 2.7413i
I2 = -0.8568 + 1.2587i
I3 = 0.8568 + 1.7413i
I1R = 2.4422 - 0.2990i
I1L = -0.2990 - 2.4422i
I2R = 0.2010 + 1.0578i
I2C = -1.0578 + 0.2010i
I3R = 1.3718 + 0.6438i
I3C = -0.5150 + 1.0975i
```

其相量图如图 8-9 所示。

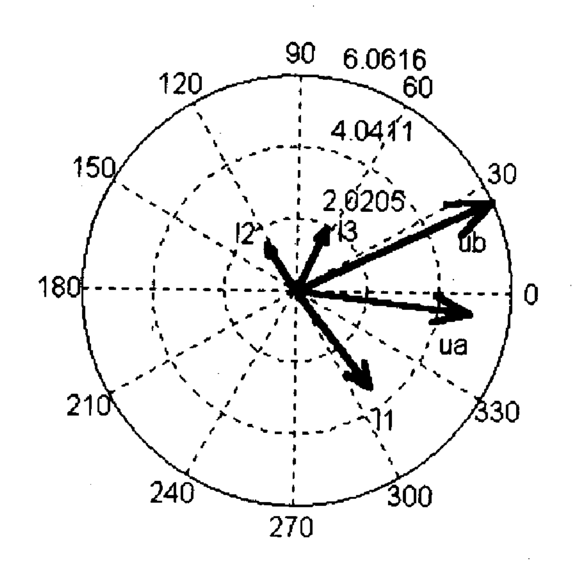


图 8-9 例 8-1-5 的相量图

【例 8-1-6】 图 8-10 为一个双电感并联单调谐网络，求回路的通频带 B 及满足回路阻抗大于 50 kΩ 的频率范围。

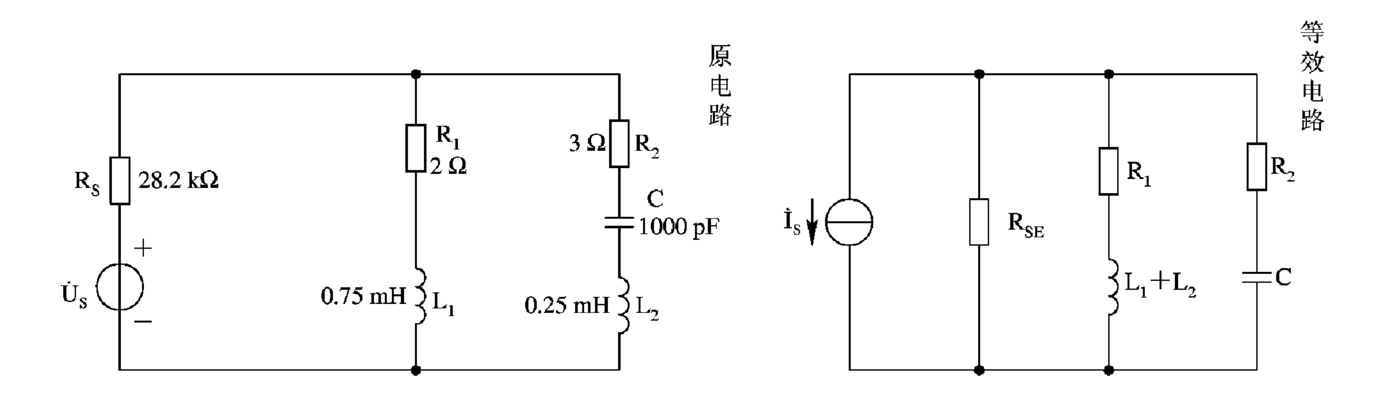


图 8-10　例 8-1-6 的电路图及等效电路

解：

◆建模

先把回路变换为一个等效单电感谐振回路，把信号源的内阻 R_S 变为并接在该单电感回路上的等效内阻 R_{SE}，如图 8-10 右图所示。按照这个等效电路可写出如下方程。

设　$m=\dfrac{L_1}{L_1+L_2}$，则

$$R_{SE}=\frac{R_S}{m^2},\ \dot{I}_S=m\frac{\dot{U}_S}{R_S}$$

其他两支路的等效阻抗分别为(设 s 为拉普拉斯算子)

$$Z_{1E}=R_1+s(L_1+L_2),\ Z_{2E}=R_2+\frac{1}{sC}$$

总阻抗是这三个支路阻抗的并联

$$Z_E=\left(\frac{1}{R_{SE}}+\frac{1}{Z_{1E}}+\frac{1}{Z_{2E}}\right)^{-1}$$

其谐振曲线可按 Z_E 的绝对值直接画出。

◆MATLAB 程序

```
r1=2; r2=3; L1=0.75e-3; L2=0.25e-3; C=1000e-12; rs=28200;
L=L1+L2; r=r1+r2; rse=rs*(L/L1)^2;    % 折算内阻
f0=1/(2*pi*sqrt(C*L))    % 谐振频率
Q0=sqrt(L/C)/r, r0=L/C/r;    % 空载(即不接信号源时)的回路 Q0 值
re=r0*rse/(r0+rse),    % 折算内阻与回路电阻的并联
Q=Q0*re/r0, B=f0/Q,    % 实际 Q 值和通带
s=log10(f0); f=logspace(s-.1, s+.1, 501);
w=2*pi*f    % 设定计算的频率范围及数组
z1e=r1+j*w*L; z2e=r2+1./(j*w*C);
                                    % 等效单回路中两个电抗支路的阻抗
ze = 1./(1./z1e+1./z2e+1./rse);    % 等效单回路中三个支路的并联阻抗
subplot(2,1,1), loglog(w,abs(ze)), grid    % 画对数幅频特性
```

```
axis([min(w), max(w), 0.9 * min(abs(ze)), 1.1 * max(abs(ze))])
subplot(2, 1, 2), semilogx(w, angle(ze) * 180/pi)    % 画相频特性
axis([min(w), max(w), -100, 100]), grid
fh=w(find(abs(1./(1./z1e+1./z2e))>5e4))/2/pi;% 幅特性大于 50 kΩ 的频带
fhmin=min(fh), fhmax=max(fh),
```

◆程序运行结果

执行此程序所得结果为：

```
谐振频率 f0 = 159.15 kHz
空载品质因数 Q0 = 200
等效信号源内阻 rse = 5.0133e+004
考虑内阻后的品质因数 Q = 40.0853
通频带 B = 3.9704e+003
回路阻抗大于 50 kΩ 的频率范围
fhmin = 157.7 kHz
fhmax = 160.63 kHz
```

谐振频率附近的幅频和相频特性曲线如图 8－11 所示。

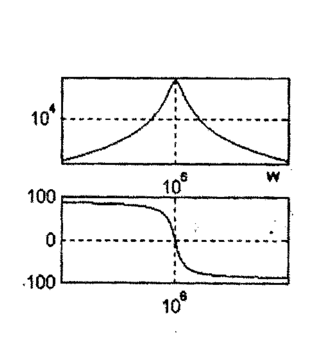


图 8－11 谐振频率处的幅频和相频特性

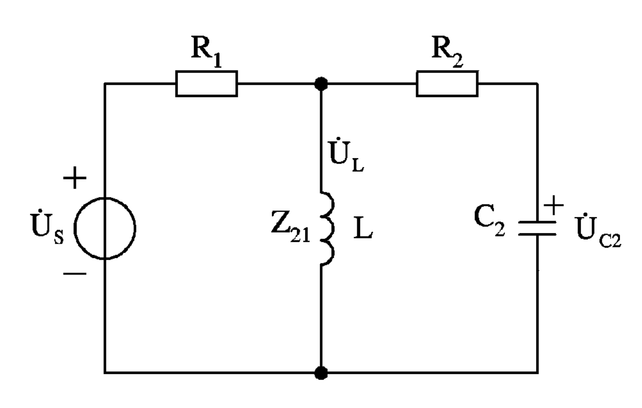


图 8－12 例 8－1－7 的电路图

【例 8－1－7】 图 8－12 所示电路中，$R_1=1\ \Omega$，$R_2=2\ \Omega$，$C_2=0.5$ F，$L=1$ H，求分别以 $\dot{U}_L$ 与 $\dot{U}_{C2}$ 为输出时的频率响应。如把 L 换成容量为 0.25 F 的电容，也求上述特性。

解：

◆建模

MATLAB 的一个优点是对同样结构的网络可以用统一的程序，如果只改变几个元件，可以用输入语句由用户来选择，我们就按这个办法来编本题的程序。

先列出方程，设 Z_{21}，Z_{C2} 分别为 L，C_2 的电抗；Z_{22} 为 R_2 与 C_2 串联的阻抗；Z_2 为 Z_{21} 与 Z_{22} 并联的阻抗。则可写出

$$\dot{U}_L=\frac{Z_2}{R_1+Z_2}\cdot\dot{U}_S$$

$$\dot{U}_{C2}=\frac{Z_{C2}}{R_2+Z_{C2}}\cdot\dot{U}_L=\frac{Z_{C2}}{Z_{22}}\cdot\dot{U}_L$$

在分析频率响应时，要假设输入信号有很多频率成分，即 ω 或 f 是一个数组，因此 MATLAB 程序中的所有与 ω 有关的量(例如 Z，$\dot{U}$，$\dot{I}$)都应采用元素群运算的运算符。

◆MATLAB 程序

```
clear, dw=0.1; w=[.2: dw: 20]; s=j*w; us=1;
r1=1; r2=2; C2=0.5; L=1; z21=s*L;
e=input('输入元件类型：电感，键入 1；电容，键入 2 ');
if e==1   L=input('输入电感量(H) '); z21=s*L;
elseif e==2   C1=input('输入电容量 (F) '); z21=1./(s*C1);
   else disp('元件类型错误，程序结束'), break, end
zc2=(1./s*C2); z22=r2+zc2; z2=z21.*z22./(z21+z22)    % 串并联计算
uL=us.*z2./(r1+z2);        % 分压计算电感上的电压
uC2=uL.*zc2./z22;          % 再分压计算电容上的电压
subplot(2, 2, 1), loglog(w, abs(uL)), grid    % 绘图，注意 subplot 用法
subplot(2, 2, 3), semilogx(w, angle(uL)), grid
subplot(2, 2, 2), loglog(w, abs(uC2)), grid
subplot(2, 2, 4), semilogx(w, angle(uC2)), grid
```

◆程序运行结果

执行此程序，输入电感为 1 H，得出如图 8－13 所示的 uL 和 uC2 的频率响应曲线。换成电容，将得到另外的结果。

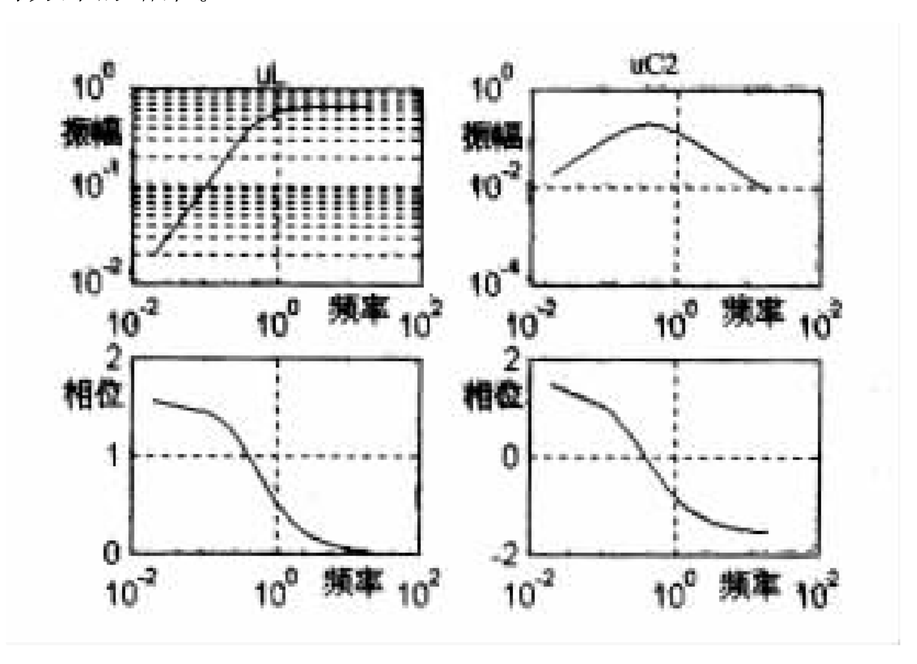


图 8－13　例 8－1－7 所示网络的频率响应

【例 8－1－8】　关于网络参数的计算。双口网络的计算公式本身并不复杂，只是公式多，其中的系数可以是复数，变量可以是相量，因此用 MATLAB 的复数矩阵计算可以带来方便，并避免出错。一个很好的办法是把这些公式列写出来，便于编程时直接拷贝调用。

解：

◆建模

1. 关于 Z，Y，A，B，H，G 等六种网络参数之间的转换关系

这里本来应该有 30 种变换关系，用 MATLAB 表示时，可以简化为三类：

(1) Z=inv(Y)；Y=inv(Z)，B=inv(A)，A=inv(B)，H=inv(G)；G=inv(H)，这三

对关系是从它们的定义得到的。

(2) A=[Z(1，1)，det(Z)；1，Z(2，2)] / Z(2，1)；

Z=[A(1，1)，det(A)；1，A(2，2)] / A(2，1)。

(3) H=[det(Z)，Z(1，2)；−Z(2，1)，1] / Z(2，2)；

Z=[det(H)，H(1，2)；−H(2，1)，1] / H(2，2)。

有了这三组关系，这六种参数中任何两者之间就都能变换了。例如已知 Y 阵，要求 G 阵，可用 Z=inv(Y)；G=inv([det(Z)，Z(1，2)；Z(2，1)，1] / Z(2，2))两个语句求得。

2. 关于网络的实验和影像参数公式

```
Zinf = A(1，1)/A(2，1)；          % Zinf——负载阻抗为∞时的输入阻抗
Zin0 = A(1，2)/A(2，2)；          % Zin0——负载阻抗为 0 时的输入阻抗
Zoutf = A(2，2)/A(2，1)；         % Zoutf——信号源阻抗为∞时的输出阻抗
Zout0 = A(1，2)/A(1，1)；         % Zout0——信号源阻抗为 0 时的输出阻抗
Zc = sqrt(Zinf * Zin0)；Zc = sqrt(Zoutf * Zout0)    % Zc——特性阻抗
gamma=20 * log10((sqrt(Zinf)+sqrt(Zin0))/ (sqrt(Zoutf)−sqrt(Zout0)))
 % gamma=u+jv 为传输常数
```

其实部 u 和虚部 v 分别为衰减常数(分贝)和相移常数(弧度)。它们都已包含在这个复数公式中，不必分别列写公式了。

现在来看一个简单的数字题，图 8－14 所示为双口网络，R=100 Ω，L=0.02 H，C=0.01 F，角频率 ω=300 1/s，求其 Y 参数及 H 参数。

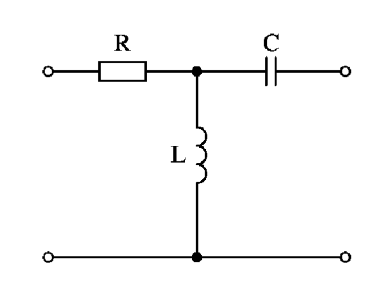


图 8－14　一个双口网络图

◆MATLAB 程序

Z 参数可以直接写出，然后求逆即可得 Y，程序为

```
r=100；L=0.02；C=0.01；w=300；
z1=r；z2 =j * w * L；z3=1/(j * w * C)；
Z(1，1) =z1+z2；Z(1，2)=z2；Z(2，1)=z2；Z(2，2)=z2+z3；
Y=inv(Z)，H=[det(Z)，Z(1，2)；−Z(2，1)，1] / Z(2，2)，
```

◆程序执行的结果

用 format long 显示为(只取小数点后 10 位)

```
Y = 0.0099998754 + 0.0000352937i        −0.0105881034−0.0000373698i
     − 0.0105881034 − 0.0000373698i       0.0112109330−0.1764310202i
H = 100.00000000 − 0.3529411765i        1.058 8235294
     1.0588235294                               0          −0.1764705882i
```

8.2　晶体管放大电路

晶体管放大电路是一门重要的课程，但是把 MATLAB 语言应用于这门课程中却比较少，可能有两个方面的原因。一是这门课中采用图解和近似估算的方法比较多，往往与各

型号的晶体管和集成电路实际特性相联系且不着重单级电路而偏重集成电路，MATLAB在这方面只有较少的用武之地；二是对于晶体管电路，包括模拟电路和数字电路，已经开发了大量的高水平 CAD 软件，可以用来进行大规模的电路设计和仿真，其中还包括了成千上万种的元器件特性可以直接调用，这些软件对于电子信息专业的人员来说，是必须在大学本科中入门的。因此，本书也有意减少了这一课程上的篇幅。

【例 8－2－1】 设将一个二极管与一电阻 R_f 串接，在此电路的两端加上正向直流电压 U_0，如图 8－15 所示，试求出此电路中的电流 I_{dx} 和电压 U_{dx}。

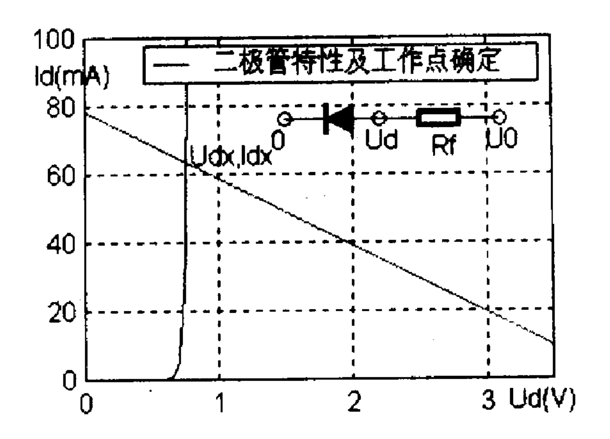


图 8－15　二极管特性和工作点的确定特性

解：

◆建模

二极管正向电流与电压的关系由下式确定

$$I_d = I_s\left[\exp\left(\frac{U_d q}{KT}-1\right)\right]$$

其中，I_s——漏电流，设为 10^{-12} A

K——玻尔茨曼常数，1.38×10^{-23}

T——绝对温度

q——电子电荷 1.6×10^{-19} C

负载电阻 R_f 中的电流 I_{d1} 随 U_d 变化的关系为

$$I_{d1}=\frac{U_0-U_d}{R_f}$$

U_d-I_d 曲线为二极管特性曲线，U_d-I_{d1} 曲线称为负载线，两者的交点 U_{dx}、I_{dx} 确定了二极管的工作点。

◆MATLAB 程序

```
% 二极管特性的计算和绘制
K=1.38e-23; T=300; q=1.6e-19;          % 给定常数
KT=K * T/q;
Is=10e-12; Ud=0:0.01:3.5;              % 给定输入电压数组
Id=Is * (exp(Ud/KT)-1);                % 求特性曲线上对应电流
plot(Ud, Id), grid on
axis([0, max(Ud), 0, 100]), hold on    % 规定绘图范围，去除太大的电流
% 线路图的绘制
line([1.5, 1.8], [79, 76])
fill([1.8, 2, 2, 1.8], [76, 72, 80, 76], 'K')   % 画二极管
line([1.8, 1.8], [72, 80], 'linewidth', 2)
line([2, 2.5], [76, 76])
line ([2.5,2.8,2.8,2.5,2.5], [74, 74, 78, 78, 74], 'line width',2) % 画电阻
line([2.8, 3.1], [76, 76])
```

```
plot([1.5, 2.2, 3.1], [76, 76, 76], 'o')
text(1.4, 70, 'o'), text(2.1, 70, 'Ud'),   % 标字符
text(2.6, 68, 'Rf'), text(3, 70, 'U0')
% 负载线的绘制
U0=input('U0=[伏]'),
Rf=input('Rf=[欧姆]')
Id1=1000*(U0-Ud)./Rf;     % 用负载线方程求 Id1[毫安]
plot(Ud, [Id; Id1]), grid on
% 寻找两曲线差为最小的点作为交点，即工作点
[di, nI]=min(abs(Id-Id1)); % 找 Id 与 Id1 数组中差为最小的元素值 dI 及序号
                            nI
Udx=Ud(nI); Idx=Id1(nI);
dsip('Udx, Idx='), [Udx, Idx], hold off
legend('二极管特性及工作点确定')    % 画图中标题
```

◆程序运行结果

运行上述程序，输入 U0=4 V，Rf=51 Ω，所得图形如图 8-15 所示。

工作点数据为 Udx=0.7600 V Idx=63.529 4 mA

可以用鼠标来求出交点的坐标，键入 ginput(1)，将鼠标游标移到图中，会出现一个十字线，将它尽可能准确地对准交点，按下鼠标左键，在命令窗中会返回该点的 Udx，Idx 值。ginput 后的参数，说明待求的点数，本例中只找一个交点，故取为 1。如果不给参数，只键入 ginput，则可以在很多点上按鼠标左键求坐标，但并不立即显示，直到按下回车键，才一起显示结果。

【例 8-2-2】 典型放大器低频等效电路如图 8-16 所示，其元件参数为

$C_1=10\ \mu F$，$R_S=100\ \Omega$，$R_b=10\ k\Omega$，

$h_{ie}=1000\ \Omega$，$h_{fe}=100$，

$R_e=200\ \Omega$，$C_e=100\ \mu F$，

$R_c=1000\ \Omega$，$C_2=10\ \mu F$，$R_L=2000\ \Omega$

要求编写其频率响应计算程序，并探讨 C_2，C_e 对幅频特性的影响。

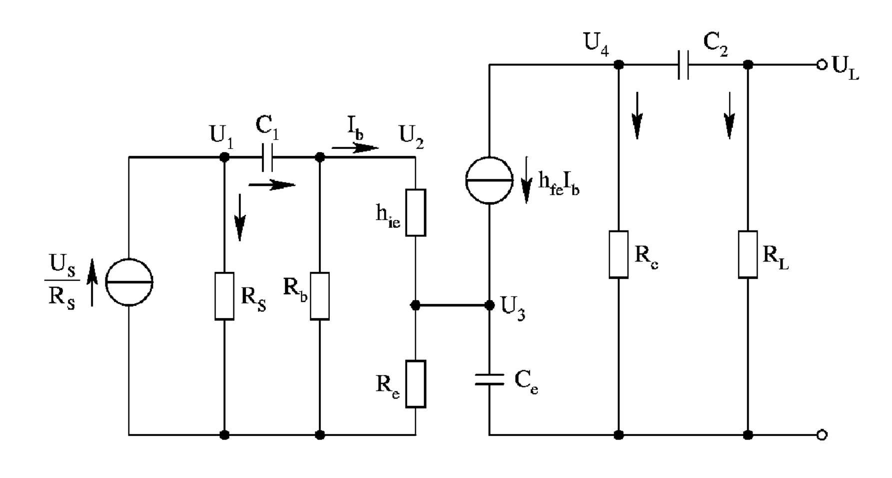


图 8-16　放大器低频等效电路

解：

◆建模

先用节点电位法列写其方程，设 U_1，U_2，U_3，U_4 如图 8－16 所示，这四个 KCL 方程如下：

$$U_S/R_S = U_1/R_S + (U_1 - U_2)sC_1 \tag{1}$$

$$(U_1 - U_2)sC_1 = U_2/R_b + (U_2 - U_3)/h_{ic} \tag{2}$$

$$(U_2 - U_3)/h_{ic} + h_{fc}(U_2 - U_3)/h_{ic} = U_3(1/R_c + sC_c) \tag{3}$$

$$h_{fe}(U_2 - U_3)/h_{ie} - U_4/R_c - U_4 sC_2/(R_L C_2 s + 1) = 0 \tag{4}$$

整理成矩阵形式(注意其中 s 为拉普拉斯算子)有

$$\begin{bmatrix} \frac{1}{R_S} + sC_1 & -sC_1 & 0 & 0 \\ -sC_1 & sC_1 + \frac{1}{R_b} + \frac{1}{h_{ic}} & -\frac{1}{h_{ic}} & 0 \\ 0 & \frac{1+h_{fc}}{h_{ic}} & -\left(\frac{1+h_{fc}}{h_{ic}} + \frac{1}{R_c} + sC_e\right) & 0 \\ 0 & \frac{h_{fe}}{h_{ic}} & -\frac{h_{fe}}{h_{ic}} & -\left(\frac{1}{R_c} + \frac{sC_2}{R_L C_2 s + 1}\right) \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} \frac{U_S}{R_S} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

写成 $a * U = b$

故 $U = a \quad b$

给定输入 $U_S=1$ 和一个频率 ω，从这个方程组就可解得输出的复数解 x，它的四个分量分别为 U_1，U_2，U_3，U_4，对于规定的频率数组作循环，即可求得其频率响应，包括振幅特性和相位特性。为了探讨 C_2 和 C_e 的影响，可在上述程序的外面，再加两个改变 C_2 和 C_e 的循环，由此编成程序如下。

◆MATLAB 程序

```
w=logspace(0，3)；   % 规定频率范围及数组值(从 100～103，按等比取 50 点)
C1=1e-5；rs=100；rb=1e4；   % 给元件赋值
hie=1000；hfe=100；us=1；
re=200；rc=1000；C2=1e-5；rL=2000；
  for C2=[C2，10*C2]      % 对 C2 及 10*C2 分别循环计算
  Ce=1e-4；
  for Ce=[Ce，10*Ce]      % 对 Ce 及 10*Ce 分别循环计算
   for i=1:length(w)    % 对各个频率计算各点输出
     s=j*w(i)；
     a11=1/rs+s*C1；a12=-s*C1；      % 给 a 矩阵元素赋值
     a21=-s*C1；a22=s*C1+1/rb+1/hie；a23=-1/hie；
     a32=(1+hfe)/hie；a33=-((1+hfe)/hie+1/re+s*Ce)；
     a42=hfe/hie；a43=-hfe/hie；a44=-(1/rc+s*C2./(rL*C2*s+1))；
     a=[a11，a12，0，0；a21，a22，a23，0；0，a32，a33，0；0，a42，a43，a44]；
```

```
        b=[us/rs; 0; 0; 0];        % 给 b 矩阵元素赋值
        x=a\b; u(:, i)=x;          % 求与第 i 个频率对应的四个输出电压
      end
      s1=j*w; uL=u(4, :).*rL./(rL+1./(C2*s1));   % 求负载电压
      loglog(w, abs(uL)), grid, hold on     % 绘对数幅频特性图
  end
end
hold off
```

◆程序执行结果

执行此程序所得的结果如图 8－17 所示，可以看出，在所给的参数下，把 C2 加大 10 倍可把低频区低端的幅频特性提高将近 10 倍，把 Ce 加大 10 倍可能在低频区的高端提高其幅频特性。

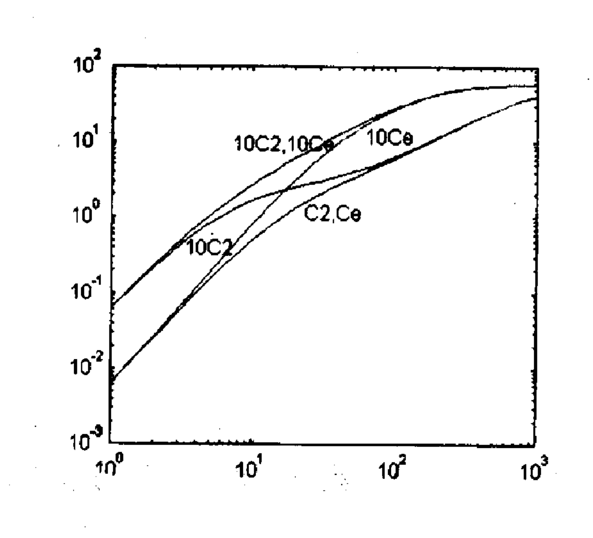


图 8－17　C2 及 Ce 对放大器低频幅频特性的影响

【例 8－2－3】　运算放大器电路如图 8－18 所示，试分析放大器开环增益和频率响应对整个电路闭环频率响应的影响，并绘出曲线。

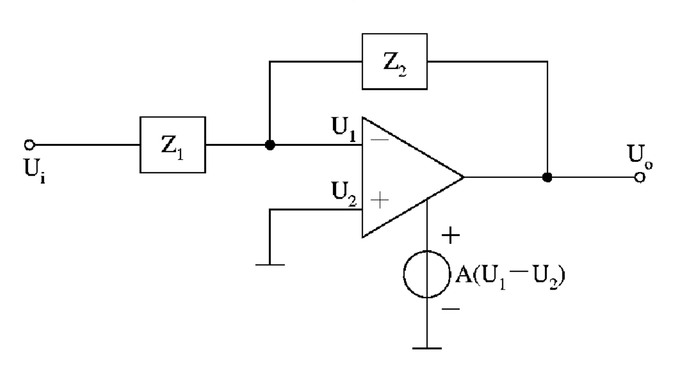


图 8－18　运算放大器等效电路

解：

◆建模

设运算放大器的开环增益为 A，它是频率的函数，则在图示的连接方法下，闭环输出与输入电压之比为

$$H=\frac{U_o}{U_i}=-\frac{Z_2/Z_1}{1+(1+Z_2/Z_1)/A} \tag{1}$$

在增益 A 很大时，分母上的第二项可以忽略不计，因而得出理想运放的闭环传递函数

$$H(s)=\frac{U_o(s)}{U_i(s)}=\frac{Z_2(s)}{Z_1(s)} \tag{2}$$

式中的 s 为拉普拉斯算子，将它换成 $j\omega$，就得出频率响应，所以这两个式子都是复数方程。根据题意，要考虑 $A=A(\omega)$ 对 $H(\omega)$ 的影响，计算将十分冗繁，利用 MATLAB 可以方便快速地解决这个问题，但必须给出具体数据。

通常，运算放大器的开环传递函数中包括三个实极点，即

$$A(s)=\frac{A_0}{\left(1+\frac{s}{\omega_1}\right)\left(1+\frac{s}{\omega_2}\right)\left(1+\frac{s}{\omega_3}\right)}=\frac{A_0\omega_1\omega_2\omega_3}{(s+\omega_1)(s+\omega_2)(s+\omega_3)}=\frac{b}{a(s)}$$

其中 $\omega_1<\omega_2<\omega_3$，取负号后为其三个极点，$A_0$ 为直流增益。

为了避免自激，通常使 ω_1 和 ω_2 差得很大，例如 $\omega_1<500$ 1/s，$\omega_2>10^6$ 1/s，而且 ω_2 和 ω_3 也要拉开，现设 $\omega_1=500$，$\omega_2=2\times10^6$，$\omega_3=5\times10^7$，并设 $Z_1=2$ kΩ，Z_2 取三种值：20 kΩ、100 kΩ 和 500 kΩ，求其 $H(\omega)$ 并绘出曲线。

◆MATLAB 程序(ex823.m)

```
% 运算放大器有限增益和频率响应对电路特性的影响
Z2=[20, 100, 500]*1000; Z1=2000;          % 设定元件参数
A0=2e6; w1=500, w2=2e6; w3=5e7;
w=logspace(2, 8);                % 设定频率数组
b=A0*w1*w2*w3;
a=poly([-w1, -w2, -w3]);      % 列出运算放大器分子分母系数向量
A=polyval(b, j*w)./polyval(a, j*w);     % 求放大器开环频率响应
for i=1:3         % 循环计算三种 Z2 的闭环响应
   Z12(i)=Z2(i)/Z1;
   H(i, :)=-Z12(i)./(1+(1+Z12(i))./A);     % 放大器闭环响应
   semilogx(w, abs(H(i, :))), hold on    % 画出频率—增益曲线
end
v=axis; axis(v);               % 保持 w 坐标
semilogx(w, abs(A))            % 画出开环频率—增益响应
hold off
```

◆程序运算结果

运行这一程序得到如图 8-19 所示的曲线，可以看出，此运放在低频区较宽的一个频带内具有平坦的增益 Z_2/Z_1。但在高频区却出现了谐振峰，这也就容易造成运算放大器的自激现象。

消除自激的方法，可以是减小 ω_1，或加大 ω_2，ω_3。因为 ω_2，ω_3 是由放大器型号的性能确定的。在放大器已经选定的情况下，通常只能用加消振电容的方法减小 ω_1，比如本题中把 ω_1 由 500 减小为 50，其他参数不变，则所得的频率响应如图 8-20 所示。可见，它大大减小了自激的可能。

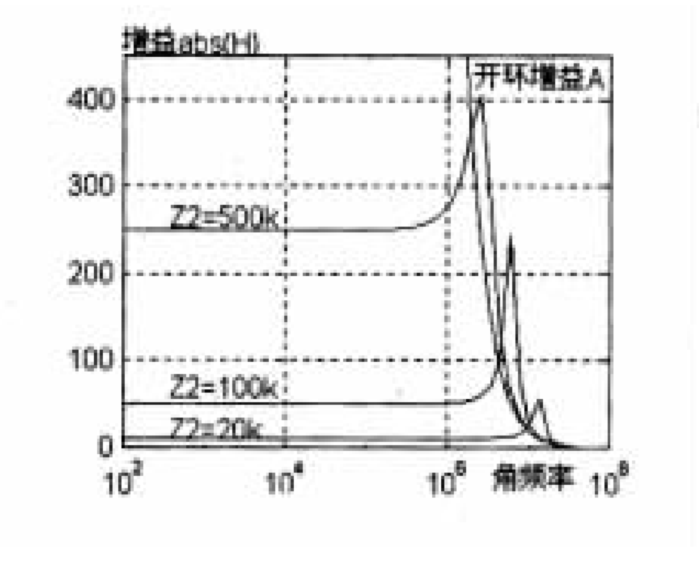


图 8-19 运算放大器的闭环频率响应

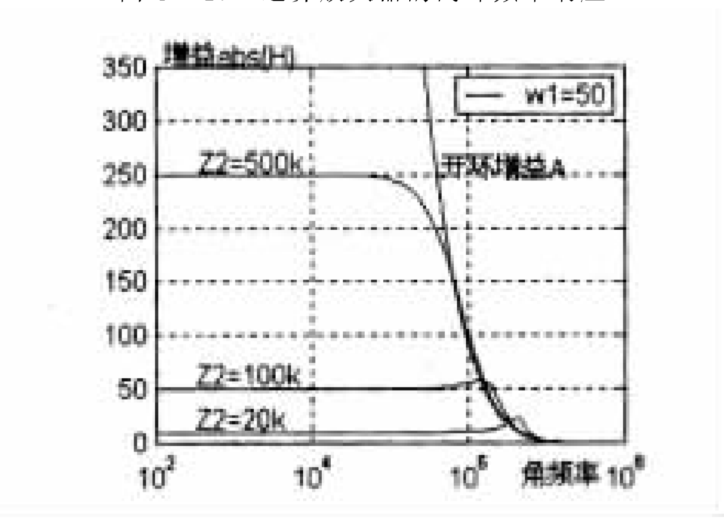


图 8-20 将 ω_1 减小 10 倍的 H(ω)

8.3 电力电子和电机

【例 8-3-1】 一个可控硅全波整流器，求出负载上得到的有效值电压与点火角(1°～180°)的函数关系，并画出曲线。

解：

◆建模

因为是全波整流，只要研究半个周波即可，其关键是如何用向量表示不连续波形。

周期为 T 的电压的有效值定义为

$$U \triangleq \sqrt{\frac{1}{T}\int_0^T u^2 dt}$$

这里用的周期是半波，即相角 $\varphi=\omega t$ 取 π 作为周期 T。将 π 分成 180 份，来表达可控硅负载电压波形，如图 8-21 所示。对应于程序中的 waveform 数组，它前一段为 0，后一段为正弦波，对该数组先进行元素群平方运算，相加平均，再做开方运算，便可得到有效值。

对不同的点火角进行循环计算，就可以画出有效值与点火角之间的关系。

◆MATLAB 程序

```
clear
wt = [1:180] * pi/180;                        % 把半个周波分割为 180 份
volts = 220 * sqrt(2) * sin(wt);              % 完整的半波波形
for ii = 1:180                                % 对不同点火角 ii 循环计算
    waveform = [zeros(1, ii-1), volts(ii:180)];   % 求不同点火角 ii 时的波形
    if ii == 45   waveform45 = waveform; end      % 记录点火角为 45°时的波形
    temp = sum(waveform.^2);                      % 计算各点波形的平方积分
    rms(ii) = sqrt(temp/180);                     % 计算积分的均方根
end
% 画负载上的有效值电压与点火角关系曲线
plot([1:180], rms, 'linewidth', 2.0);
% 下一条语句中的“ bf”表示用粗体字
legend(' bf 负载电压有效值(v)');
gtext(' bf 点火角(度)');
grid on; pause
% 画点火角 45°时负载上的电压波形
figure(2)
plot(wt, waveform45, 'linewidth', 2.0);
% 下一条语句中的 omega 和 itt 说明如何标注希腊字母 ω 和 t
gtext(' bf  omega  itt  rm 弧度');
grid on;
```

◆程序运行结果

执行这个程序所得的结果如图 8－21 及图 8－22 所示。

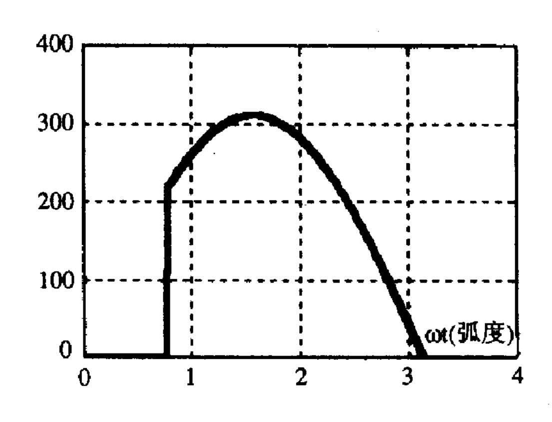


图 8－21　点火角 45°时的电压波形

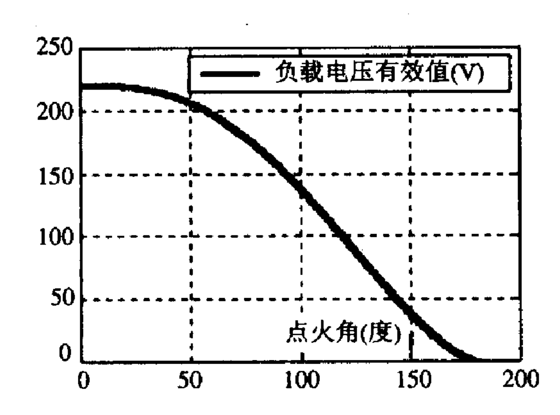


图 8－22　电压有效值与点火角关系

【例 8－3－2】　用 MATLAB 语言演示感应电动机三相定子磁场的合成。说明其磁场矢量如何合成为旋转磁场，程序具有一定的动画效果，读者可从中学习到动画程序编写的一些技巧。

解：

◆建模

感应电动机的定子由三组空间上相差120°的绕组和磁极组成，如图8-23的aa′，bb′，cc′所示，在三个绕组上依次加有120°相位差的励磁电压，就可以形成一个在空间旋转的磁场。

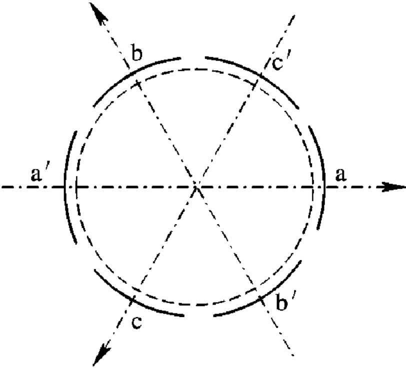


图8-23　三相磁极

利用MATLAB来演示这一过程特别有效：

(1) 利用MATLAB的复数功能，描述三个磁极在空间不同方向产生的磁场。

(2) 利用时间数组来描述三个绕组中电流和磁场的相位变化。

(3) 把三个磁场作向量相加，即可求得合成磁场。

(4) 利用MATLAB的绘图和动画功能显示磁场的运动。

根据这个思路来编写MATLAB程序较为容易。

◆MATLAB程序

```
clear, clf
I=10; freq = 50; w = 2 * pi * freq;        % 50 Hz 角频率(rad/s)
t = 0: 1/5000:2/50;
% a, b, c 相位差为120°的三相电流
Ia=I * sin(w * t); Ib=I * sin(w * t-2 * pi/3); Ic=I * sin(w * t+2 * pi/3);
% 建立三个磁场分量的表达式
kmag = 1/I;   % 选适当的绕组常数，把最大磁场归一化为1
Baa = kmag * Ia * (cos(0) + j * sin(0)); % a磁场空间方向为0°
Bbb = kmag * Ib * (cos(2 * pi/3) + j * sin(2 * pi/3)); % b磁场方向为pi/3
Bcc = kmag * Ic * (cos(-2 * pi/3) + j * sin(-2 * pi/3)); % c磁场方向为-pi/3
Bnet = Baa + Bbb + Bcc;        % 计算合成磁场
% 分别画出合成磁场Bnet和三相磁场Baa，Bbb，Bcc的矢量幅值和方向
% Bnet为红色，Baa为黑色，Bbb为蓝色，Bcc为品红色
for ii = 1: length(t)
    plot(Bnet, 'k');    % 画出合成磁场向量端点的轨迹，它是一个圆
    hold on;
    % 画出四个磁场相量,前三个方向固定,大小随时间变化,第四个为合成磁场
    plot([0 real(Baa(ii))], [0 imag(Baa(ii))], 'k', 'LineWidth', 2);
    plot([0 real(Bbb(ii))], [0 imag(Bbb(ii))], 'b', 'LineWidth', 2);
    plot([0 real(Bcc(ii))], [0 imag(Bcc(ii))], 'm', 'LineWidth', 2);
    plot([0 real(Bnet(ii))], [0 imag(Bnet(ii))], 'r', 'LineWidth', 3);
    axis square; axis([-2, 2, -2, 2]); drawnow;
    hold off;
end
```

◆程序运行结果

执行此程序，将得到一个演示旋转磁场的动画，图 8－24 给出了其中一个画面，Baa，Bbb，Bcc 表示三个方向固定，在空间上夹角为 120°的磁场，它们的大小和正负号按交流电流的正弦波变化，相位互差 120°，三者的向量和就形成了一个在空间旋转的磁场。

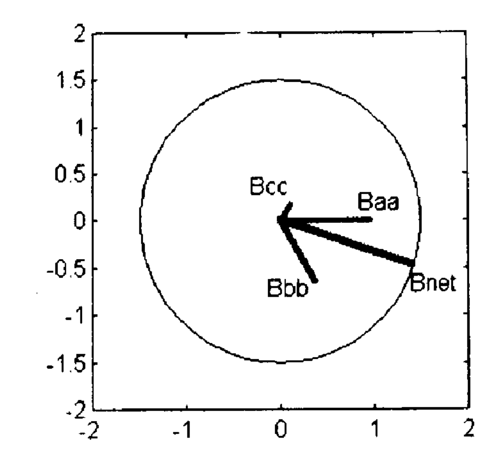


图 8－24　三相交变磁场合成旋转磁场
（三个方向相差 120°的磁场向量和为 Bnet）

在程序中要注意 drawnow 的用法，通常 MATLAB 采用先计算，后画图的顺序，在程序中即使出现画图命令，它也只把数据存起来，作好画图准备，继续进行计算，直到程序暂停或结束时才统一画图。在显示动画时，这种做法就不行了，必须算出后立即显示，故要加 drawnow 命令。但程序执行的速度将大大下降。在编动画程序时还要注意的是 hold on 和 hold off 命令放置的位置。

这个动画运行的效果不太好。主要因为每一次更新矢量时，也要更新所有的坐标，不仅速度慢，而且图形跳动，使观察者眼睛不舒服。改进的方法是只更新矢量，其余部分不动。这就要用低层图形命令，只改变几个运动图形对象的图柄数据。本书附盘中提供的程序 ex832a.m 就照这个办法编写，其效果非常好，读者可自行运行这个程序并分析其程序的编写方法。

【例 8－3－3】　感应电动机机械特性曲线绘制。

解：

◆建模

感应电动机中任一相的等效电路如图 8－25(*a*)，其中 S 为转差率。再用戴文宁定理依次等效为图 8－25(*b*)和(*c*)，根据此图列写方程如下：

$$U_{ab}=U_{ph}\cdot Z_m/(Z_1+Z_m)$$

$$Z_{ab}=Z_mZ_1/(Z_1+Z_m)$$

由最后的等效电路可得

$$R_{2c}=R_2+\frac{(1-S)}{S}R_2=\frac{1}{S}R_2$$

$$I_{2e}=U_{ab}/(Z_{ab}+R_{2e}+jX_2)$$

消耗在转子回路中的总功率(三相功率总和)

$$P_{2c}=3|I_{2c}|^2\cdot R_{2c}$$

电磁转矩与同步角速度的乘积等于电磁场功率，故有

$$torque=P_{2c}/wsync$$

◆MATLAB 程序

```
%程序中尽量直接用复数计算阻抗，并利用元素群计算以简化程序
clear, fvolt=50;                          % 电源频率
r1 = 0.641; x1 = 1.106; z1 = r1 + j * x1;  % 定子相绕组电阻和电抗
r2 = 0.332; x2 = 0.464; z2 = r2 + j * x2;  % 转子等效电阻和电抗
```

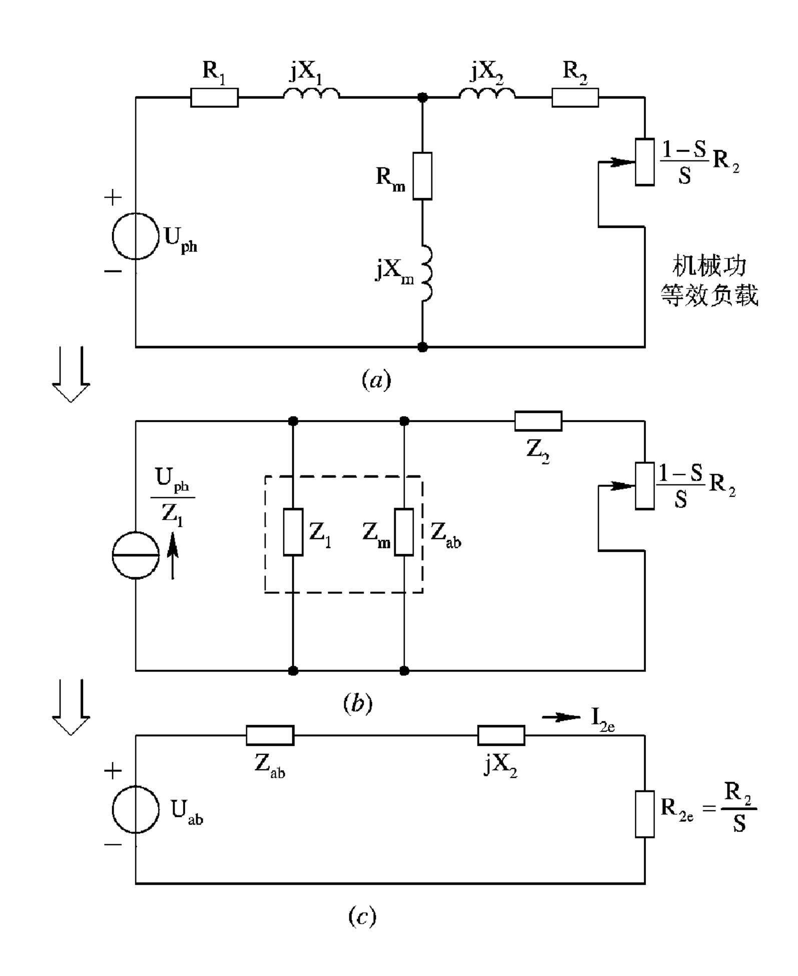


图 8-25 感应电动机等效电路

```
rm = 2.5; xm = 26.3; zm = rm + j * xm;  % 磁化支路等效电阻和电抗
uph = 380 / sqrt(3);            % 相电压 (u)
p=2, nsync = fvolt/p * 60;      % 同步转速 (r/min)(p=2, 双极电机)
wsync = nsync * 2 * pi/ 60;     % 同步角速度 (rad/s)
% 计算 A, B 两点之间电压 uab 和阻抗 zab
uab = uph * zm/(z1+zm);
zab = zm * z1 / (z1+zm);
for r2 =[r2, 2 * r2]      % 求给定内阻及内阻加倍两种电机转矩与转速关系
  s = (eps: 1: 50) / 50;              % 转差率数组, 第一点避开 0
  nm = (1 - s) * nsync;               % 电机转速数组
  z2e = r2./s + j * x2;               % 转子等效负载
  I2e = uab ./(zab+z2e);              % 转子等效电流
  P2e = 3 * abs(I2e.^2 * r2./s);      % 三相电磁功率总和
  torque = P2e / wsync;               % 转矩等于电磁功率除以同步角速度
  % 画机械特性曲线
  plot(nm, torque, 'LineWidth', 2.0);
  hold on;
end
xlabel('转速'), ylabel('转矩'),
```

```
gtext ('感应电机转矩转速特性'),
grid on;
```

◆程序运行结果

程序运行后，将得出如图 8－26 所示的转矩转速特性。

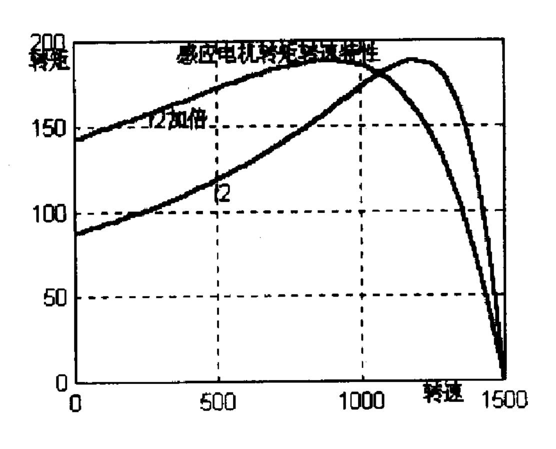


图 8－26　感应电机转矩转速特性

8.4　高 频 电 路

【例 8－4－1】　设无损耗同轴线内导体半径为 a，外导体内半径为 b，内外导体间的媒质参数为(ε，μ)，求其特性阻抗及绘出它随 ε 变化的曲线。

解：

◆建模

根据电磁场理论，无损耗同轴线单位长度的电感和电容分别为

$$L=\frac{\mu}{2\pi}\ln\frac{b}{a},\qquad C=\frac{2\pi\varepsilon}{\ln\frac{b}{a}}$$

故其特性阻抗为

$$Z_0=\sqrt{\frac{L}{C}}=\frac{1}{2\pi}\sqrt{\frac{\mu}{\varepsilon}}\ln\frac{b}{a}$$

通常同轴线内外导体间的媒质为绝缘电介质(例如聚乙烯、聚四氟乙烯)，其参数为 $\mu=\mu_0$，$\varepsilon=\varepsilon_0\varepsilon_r$，其中 ε_r 为相对介电常数，因为 $\mu_0/\varepsilon_0=1.42\times10^5$，故得

$$Z_0=\frac{60}{\sqrt{\varepsilon_r}}\ln\frac{b}{a}$$

◆MATLAB 程序

```
% 取常用于传输有线电视信号的同轴线为例，设 a=0.5 mm，b=4 mm
a=0.5; b=4;
% 设定相对介质常数范围
er=linspace(1,20);
z0=60*log(4/0.5)./sqrt(er);
% 特性阻抗
```

```
plot(er, z0);
grid
xlabel('相对介电常数');
legend('特性阻抗(欧姆)')
```

◆程序运行结果

程序运行的结果如图 8－27 所示。通常电视传输同轴线的特性阻抗标准为 50 Ω，由此可确定要求的介质常数。

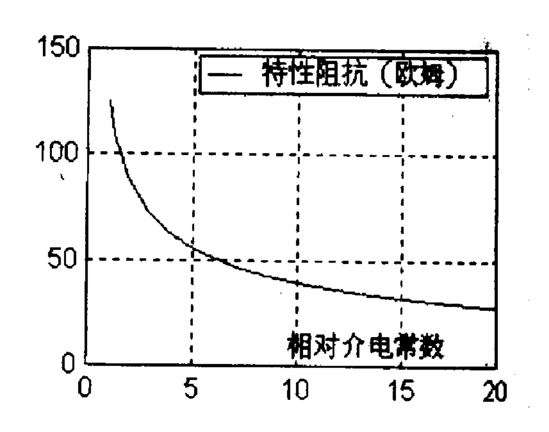


图 8－27 同轴电缆特性阻抗与介质常数的关系

【例 8－4－2】 用 MATLAB 语言画 Smith 图。

解：

◆建模

Smith 图是在高频工程中常常用到的。它建立了归一化阻抗 z 和复反射系数 γ 之间的图形关系。

两者间的数学关系为

$$\gamma = (z-1)/(z+1) = u + iv$$

其中，z=r+i＊x。

绘图时的横坐标是 γ 的实部 u，纵坐标是它的虚部 v。指定参数 r=常数及 x=常数，得到圆的轨迹，即 Smith 图。由此图可以根据 γ 来求 r 及 x，也可由 r 及 x 求得反射系数 u 和 v。为了绘制这个图，可先把上面的复数方程化为两个实数方程，其结果为：

$$v^2 + \left(u - \frac{r}{1+r}\right)^2 = \frac{1}{(1+r)^2} \qquad \text{固定 r，变化 x 时(消去 x)的轨迹}$$

$$\left(v \pm \frac{1}{x}\right)^2 + (u-1)^2 = \frac{1}{x^2} \qquad \text{固定 x，变化 r 时的轨迹}$$

这两个都是圆的方程，可找出其圆心及半径，以圆心角为参数画出圆。

◆MATLAB 程序

```
plot([0 0], [-1.1 +1.1], 'r'), hold on, xlabel('u')   % 画坐标轴系
plot([-1.1 +1.1], [0 0], 'r'), ylabel('v'),
axis equal, axis([-1.1, 1.1, -1.1, 1.1]), grid
tr = 2*pi*(0:.01:1);          % 指定所要画的圆的圆周角分度数组
for r = [0, .2, .5, 1, 2, 5]                  % 指定所要画的圆的参数 r
   rr = 1/(r+1); cr = 1-rr;
   plot(cr+rr*cos(tr), rr*sin(tr))            % 画等 r 圆
end
for x = [.2, .5, 1, 2, 5]                     % 指定所要画的圆的参数 x
   rx = 1/x; cx = rx;
   tx = 2*atan(x)*(0:.01:1);                  % 等 x 圆的分度数组
   plot(1-rx*sin(tx), cx-rx*cos(tx))          % 画等 x 圆，x>0
   plot(1-rx*sin(tx), -cx+rx*cos(tx))% 画等 x 圆，x<0
```

```
end
```

◆程序执行结果

执行程序的结果如图 8－28 所示。其上的参数可用 gtext 命令由鼠标标注，免去确定坐标的麻烦。

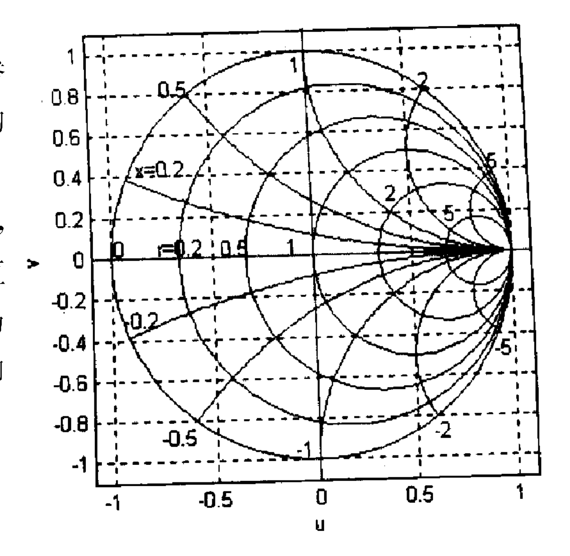


图 8－28　Smith 图

实际上，利用 MATLAB 的复数运算功能，也可以省去把复数方程化为实数方程的推导。直接用上述复数方程，分别设定参数 r 和 x 之一为常数，另一个为数组，按 γ 的实部和虚部绘图即可。其核心的语句是：

```
z=(r+i*x-1)./(r+i*x+1);
plot(z)
```

其主要难度是 r 和 x 数组不太好设定，设得不好，画出的圆会疏密不匀。读者可参看下列 MATLAB 程序(ex842a.m)，这个程序中的数组 q 就是精心选择的。

```
p = logspace(-2, 2); x=[-p, p];
q = [eps, 0.05, 0.10.2, 0.5, 1, 2, 5, 10];
for r = q
   z=(r+i*x-1)./(r+i*x+1);
   plot(z), hold on, axis equal
end
r = p;
for x=[-q, q]
   z=(r+i*x-1)./(r+i*x+1);
   plot(z), hold on, axis equal
end
grid, hold off
```

运行此程序得的曲线如图 8－29 所示。

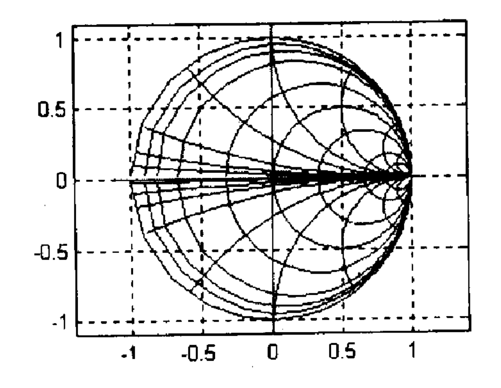
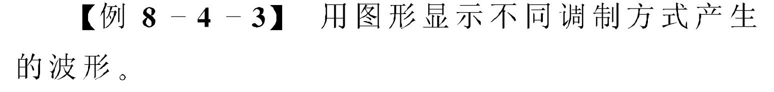


图 8－29　用复数方程画的 Smith 图



【例 8－4－3】　用图形显示不同调制方式产生的波形。

解：

◆建模

调幅、调频及调相信号的波形可用同一个 MATLAB 程序产生；只要在输入参数时分别给'振幅增量='、'频偏='和'相移='之一赋值，而令其余两个为零即可。

一般的调制波方程为

$$y=(A_0+\Delta A)\sin((\omega_0+\Delta\omega)t+\Delta\varphi)$$

ω_0 为载频，使它的振幅增量 ΔA、频率增量 $\Delta\omega$ 和相位增量 $\Delta\varphi$ 分别随要传送的信号变化，就形成调幅、调频或调相。要传送的信号频率，通常都远低于载频。为了便于比较，这

里信号的频率为 $0.1\omega_0$，分别单独调制振幅、频率和相位，以便观察它们在时间曲线、频谱特性方面的差别。

◆MATLAB 程序

```
clf, format compact, t=0: 0.001: 1;
A0=10, w0=100, phi0=0
x=A0. * sin(w0 * t+phi0); subplot(3, 1, 1), plot(t, x), hold
xlabel('t'), ylabel('载频信号')
dA=input('振幅增量='); dw=input('频偏='); dphi=input('相移=');
y=(10+dA). * sin((w0+dw). * t+dphi);
subplot(3, 1, 2),
plot(t, y), ylabel('调制波形')
subplot(3, 1, 3),
Y=fft(y); plot(abs(Y)), grid   % 求 fft(y)并画出其振幅特性
axis([0, 50, 0, 5000])   % 展宽重要的频段图形
```

◆程序运行结果

运行此程序，输入三种不同参数，即可得三种不同调制方式的波形分别如图 8－30、图 8－31 和图 8－32 所示。

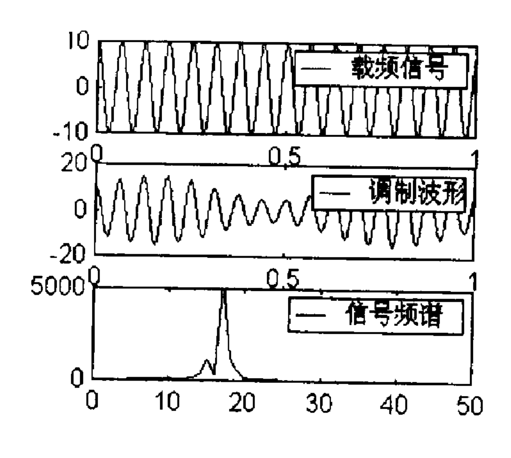


图 8－30　调幅波形

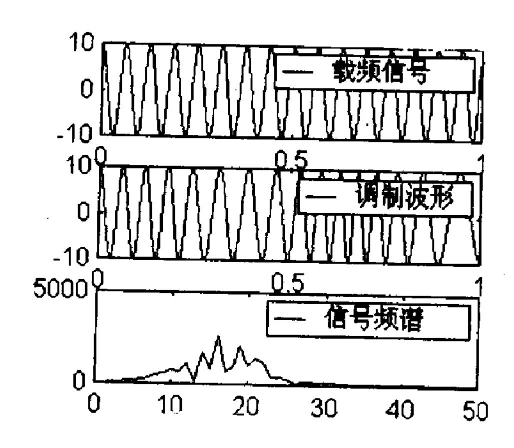


图 8－31　调频波形

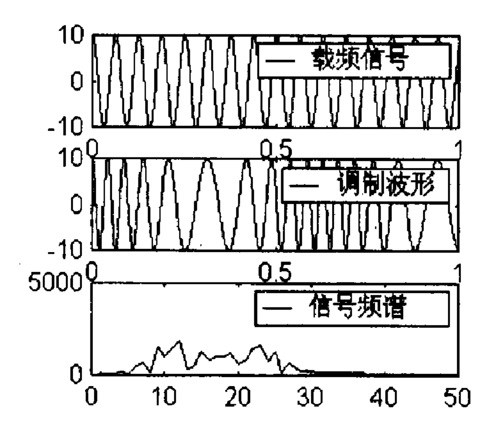


图 8－32　调相波形

A0 =10，w0 =100，phi0 = 0

(1) 调幅。

振幅增量=5 * sin(0.1 * w0 * t)

频偏=0

相移=0

(2) 调频。

振幅增量=0

频偏=5 * sin(0.1 * w0 * t)

相移=0

(3) 调相。

振幅增量=0

频偏=0

相移=5 * sin(0.1 * w0 * t)

第 9 章　在信号和系统中的应用举例

9.1　连续信号和系统

【例 9－1－1】 列出单位脉冲、单位阶跃、复指数函数等连续信号的 MATLAB 表达式。

解：

◆建模

严格说来，MATLAB 是不能表示连续信号的，因为它给出的是各个样本点的数据，只有当样本点取得很密时可看成连续信号。密，要相对于信号变化的快慢而言，形象地说，在所有相邻样本点之间的数据变化必须非常小才能看成“密”，其严格的数学定义此处不予讨论。在本题中约定，相对于采样点密度而言，信号变化足够慢。在编程中，先设定共同的时间坐标，然后分别列出生成三种信号的程序。

◆MATLAB 程序

```
clear, t0=0; tf=5; dt=0.05; t1=1; t=[t0: dt: tf];
% (1) 单位脉冲信号
% 在 t1(t0≤t1≤tf)处有一持续时间为 dt，面积为 1 的脉冲信号，其余时间均为零
  t=[t0: dt: tf]; st=length(t);
  n1=floor((t1-t0)/dt);            % 求 t1 对应的样本序号
  x1=zeros(1, st);                 % 把全部信号先初始化为零
  x1(n1)=1/dt;                     % 给出 t1 处的脉冲信号
  subplot(2, 2, 1), stairs(t, x1)  % 绘图，注意为何用 stairs 而不用 plot 命令
  axis([0, 5, 0, 1.1/dt])          % 为了使脉冲顶部避开图框，改变图框坐标
% (2) 单位阶跃信号
% 信号从 t0 到 tf，在 t1(t0≤t1≤tf) 前为 0，到 t1 处有一跃变，以后为 1
  x2 = [zeros(1, n1-1), ones(1, st-n1+1)];  %产生阶跃信号
  subplot(2, 2, 3), stairs(t, x2)           % 绘图
  axis([0, 5, 0, 1.1])   % 为了使方波顶部避开图框，改变图框坐标
% (3) 复数指数信号
  u=-0.5; w=10; x3=exp((u+j*w)*t);
  subplot(2, 2, 2), plot(t, real(x3))       % 绘图
  subplot(2, 2, 4), plot(t, imag(x3))       % 绘图
```

◆程序运行结果

x1、x2、x3 的波形如图 9－1 所示。注意：若要显示连续信号波形中的不连续点，用 stairs 命令；若要使波形光滑些，则用 plot 命令较好。复数指数信号 x3 可以分解为余弦和正弦信号，它们分别是复数信号的实部和虚部，右图中的两个衰减振荡信号就代表了这两个相位差 90°的分量。

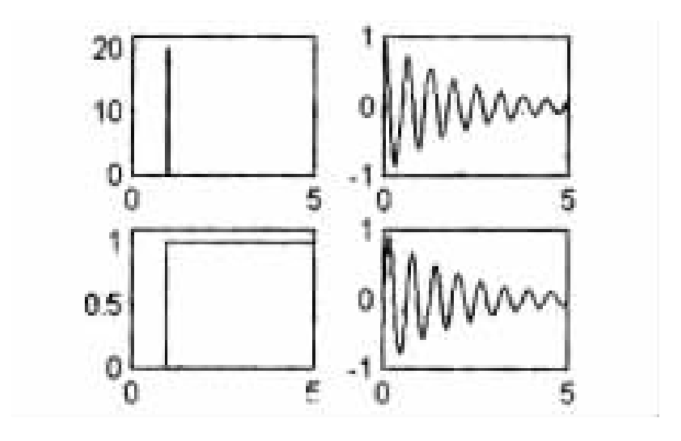


图 9－1　例 9－1－1 中 x1、x2、x3 对应的四种波形

【例 9－1－2】　设二阶连续系统，其特性可用常微分方程表示

$$\frac{d^2x}{dt^2}+3\frac{dx}{dt}+6=u$$

求其冲击响应。若设输入为 u=3t，求系统的输出。

解：

◆建模

先求系统的响应。冲击信号只造成一个初始状态，以后输入为零，因此，它相当于零输入情况，写出其特征方程

$$\lambda^2+3\lambda+6=0$$

其特征根为 p_1、p_2，则冲击响应为

$$h(t)=r_1e^{p_1t}+r_2e^{p_2t}$$

其中，r_1、r_2 为对应于 p_1、p_2 的留数（r_1、r_2 也可用初始条件来确定，见例 9－1－3）。输出 x(t) 可用输入 u(t)与冲击响应 h(t)的卷积求得

$$x(t)=\int_0^t u(\tau)h(t-\tau)d\tau$$

可以按这个式子编写 MATLAB 程序。解二阶系统固有运动的程序及其结果可参阅例 7－3－1。但这种按照最后公式编写程序的方法并不高明，因为要进行很多手工推导，容易出错，最好让 MATLAB 帮助我们做推导公式中的运算。此时可参照 4.3 节中用留数解微分方程的方法，从而得到求冲击响应的程序如下。

◆MATLAB 程序

```
a=[1, 3, 6]; b=2; c=3; dt=0.1; tf =5;
t=0:dt:tf;      % 设定自变量数组
[r, p] = residue( b, a);         % 求系统的极点留数
h=r(1) * exp(p(1) * t)+r(2) * exp(p(2) * t);
subplot(2, 1, 1), plot(t, h); grid
```

```
%把输入信号与冲击响应 h 求卷积
u=c*t; x=conv(u, h)*dt;
subplot(2, 1, 2), plot(t, x(1: length(t)), grid
```

◆程序运行结果

如图 9－2 所示，上图表示冲击响应 h，下图表示输出波形。

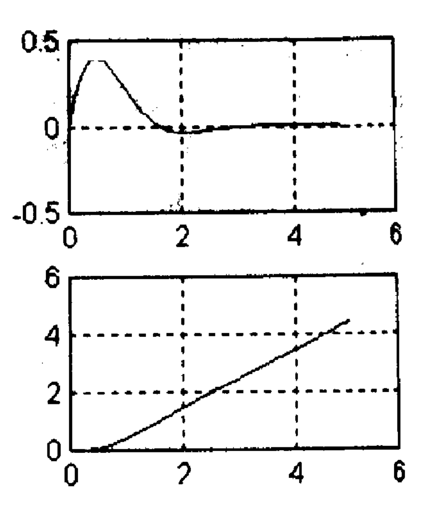


图 9－2 冲击响应和卷积法求输出

程序的最后两行可求给定输入下的输出。这里采用卷积函数来求，即线性系统的输出等于输入信号和系统冲击响应的卷积。注意卷积的输出序列长度为 u 和 h 长度之和减 1。在程序执行后用 whos 命令检查，可知 x 的长度为 101，t 的长度为 51，故最后一句绘图命令必须限定只取 x 中的前面 51 个点。读者可自行分析 x 的后 50 个点具有何种意义。

【例 9－1－3】 编写求任意高阶连续线性系统冲击响应的程序。

解：

◆建模

任意阶次的连续线性系统可用下列线性常微分方程表述

$$a_1\frac{d^n y}{dt^n}+a_2\frac{d^{n-1}y}{dt^{n-1}}+\cdots+a_n\frac{dy}{dt}+a_{n-1}=b_1\frac{d^m u}{dt^m}+\cdots+b_m\frac{du}{dt}+b_{m+1}\qquad n\geqslant m$$

写成传递函数形式为

$$H(s)=\frac{Y(s)}{U(s)}=\frac{b_1s^m+b_2s^{m-1}+\cdots+b_ms+b_{m-1}}{a_1s^n+a_2s^{n-1}+\cdots+a_ns+a_{n+1}}$$

因此，其特性可以用系统传递函数的分子、分母系数向量 b 和 a 来表示，对于一切物理上可实现的系统，length(a)≥length(b)，即 a 的长度不会比 b 小。length(a)－1 就等于系统的阶次。如果系统的分母部分没有重根，则其冲击响应可用下列程序来求。

◆MATLAB 程序(ex913.m)

```
a=input('多项式分母系数向量 a= ')
b=input('多项式分子系数向量 b= ')
[r, p] = residue( b, a),    % 求留数
disp('冲击响应的解析式为 h(t)=Σr(i)*exp(p(i)*t)')
k=input('是否要求波形？是，键入 1；否，键入 0 ');
if k==1
   dt=input('dt= '); tf=input('tf= ');     % 设定时间数组
   t=0: dt: tf; h=zeros(1, length(t));     % h 的初始化
   for i=1: length(a)-1                    % 根的数目等于 a 的长度减 1
        h = h+ r(i)*exp(p(i)*t);           % 叠加各根分量
   end
   plot(t, h), grid
else,   end
```

◆程序运行结果

例如，给出系统传递函数为

$$H(s)=\frac{s^2+7s+1}{s(s+1)(s+2)(s+5)}$$

求冲击响应。

键入 ex913，根据程序提问依次输入：

```
a=poly([0，-1，-2，-5])
b=[1，7，1]
dt=0.05
tf=5
```

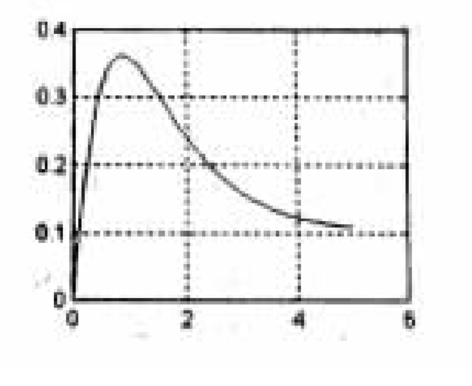


图 9－3　高阶系统的冲击响应

得出的 h(t)如图 9－3 所示。

程序中要的是系数向量 a，而题中给出的是极点向量 p=[0，-1，-2，-5]，因此这里用 poly 函数来作转换。

【例 9－1－4】 对于任意线性时不变连续系统，其特性可用常微分方程表示：

$$a_1\frac{d^n y}{dt^n}+a_2\frac{d^{n-1}y}{dt^{n-1}}+\cdots+a_n\frac{dy}{dt}+a_{n+1}=b_1\frac{d^m u}{dt^m}+\cdots+b_m\frac{du}{dt}+b_{m+1}\qquad n\geqslant m$$

求输入 u 为 0 时，由初始状态决定的输出，即其零输入响应。

解：

◆建模

在零输入条件下，系统的响应取决于微分方程左端特征方程的根，与右端无关，其通解为

$$y=C_1e^{p_1t}+C_2e^{p_2t}+\cdots+C_ne^{p_nt}$$

其中，p_1、p_2、…、p_n 是特征方程 $a_1\lambda^n+a_2\lambda^{n-1}+\cdots+a_n\lambda+a_{n+1}=0$ 的根。每个分量的系数 C_1、C_2、…、C_n，应由 y 及其各阶导数的初始条件来确定。

$$y|_{t=0}=C_1+C_2+\cdots+C_n=y_0$$

$$\dot{y}|_{t=0}=p_1C_1+p_2C_2+\cdots+p_nC_n=Dy_0$$　（Dy_0 表示 y 的导数的初始值）

$$\ddot{y}|_{t=0}=p_1^2C_1+p_2^2C_2+\cdots+p_n^2C_n=D^2y_0$$

…

初始条件数应该和待定系数相等，构成一个确定 C_1、…、C_n 的线性代数方程组，写成

$$V\cdot C=Y_0$$

其解为

$$C=V\quad Y_0$$

其中

$$C=[C_1, C_2, \cdots, C_n]';\ Y_0=[y_0, Dy_0, \cdots, D^ny_0]'$$

$$V=\begin{bmatrix}1 & 1 & \cdots & 1\\ p_1 & p_2 & \cdots & p_n\\ p_1^2 & p_2^2 & \cdots & p_n^2\\ \cdots & \cdots & \cdots & \cdots\\ p_1^{n-1} & p_2^{n-1} & \cdots & p_n^{n-1}\end{bmatrix}$$

这种形式的矩阵称为范德蒙特矩阵。在 MATLAB 的特殊矩阵库中提供了专门的函数 vander，给出 p 向量就可由 V=vander(p)生成范德蒙特矩阵。从 help vander 知道，需要将

它旋转 90°，才与本题的形式相符。

◆MATLAB 程序

```
a=input('输入分母系数向量 a=[a1, a2, ...]= ');
n=length(a)-1;
Y0=input('输入初始条件向量 Y0=[y0, Dy0, D2y0, ...]= ');
p=roots(a);V=rot90(vander(p));C=V\Y0';% 构成所需的范德蒙特矩阵，求出C
dt=input('dt='); tf=input('tf= ')
t=0: dt: tf; y=zeros(1, length(t));          % 给出自变量数据组
for k=1: n y= y+c(k) * exp(p(k) * t); end    % 求各分量的时间函数并叠加
plot(t, y), grid
```

下面利用这一程序来解一个三阶系统。

◆程序运行结果

运行此程序并输入：

a=[1, 2, 9, 3];

dt=0.1; tf=5;

而初始值 Y0 分别取

[1, 0, 0]; [0, 1, 0]; [0, 0, 1]

用 hold on 语句使三次运行生成的图形画在一幅图上，得到图 9-4。

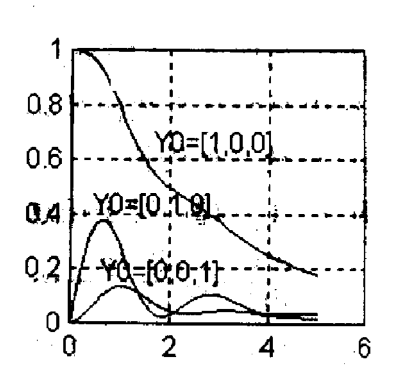


图 9-4 三阶系统零输入分量的解

【例 9-1-5】 n 级放大器，每级的转移函数均为 $\frac{\omega_n}{s+\omega_n}$，求阶跃输入下的过渡过程，画出 n 不同时的波形及频率特性进行比较。

解：

◆建模

系统的转移函数为 $H(s)=\frac{\omega_n^n}{(s+\omega_n)^n}$，阶跃输入的拉普拉斯变换为 $\frac{1}{s}$，因此输出为两者的乘积，即

$$Y(s)=\frac{\omega_n^n}{s(s+\omega_n)^n}$$

求 Y(s)的拉普拉斯反变换，即可得到输出过渡过程 y(t)。

这里我们遇到了一个有多重极点 $-\omega_n$ 的 H(s)求拉普拉斯反变换的问题。从原理上说，同样可以用 4.3 节中的留数极点分解法来求。只是在有 n 重根时，分解出的部分分式的分母将不再是一次极点，而有$(s+\omega_n)$、…、$(s+\omega_n)^{n-1}$、$(s+\omega_n)^n$ 等项，而$(s+\omega_n)^{-q}$的反变换可用如下解析式表示

$$h(t)=L^{-1}\left[\left(\frac{\omega_n}{s+\omega_n}\right)^q\right]=\frac{1}{(q-1)!}t^{q-1}(e^{-\omega_n t}-1)$$

按照这个思路，应该先求出 Y(s)的极点留数。注意分母中除了有 n 个重极点外还有一个零极点(即 1/s)，故共有 n+1 个极点。先用 poly 函数求出 Y(s)分母多项式的系数向量，并求出极点及相应的留数：

```
    by=wn^n; ay=[poly(-ones(1, n) * wn), 0]
    [r, p] = residue(by, ay);
```

再把各个分量叠加在一起。然而，实际上，这样编程不仅非常麻烦，而且难以得到正确的结果。其原因是在重极点处，MATLAB 的 residue 算法遇到了病态问题，数据中小小的舍入误差会使结果产生很大误差，即使 n 取 2 都得不出正确结果。

为了避开重极点问题，可以有意把极点拉开一些。例如，设 n 个极点散布在 $0.98\omega_n \sim 1.02\omega_n$ 之间，那样就可以全部作为非重极点来列程序。这种处理在工程上是完全没有问题的，一般电阻的标称误差为±5%，电容则更大，在工程实践中，使各个放大器时常数完全相同是不可能的，即便要把其误差控制到±2%以内也非易事。由此在工程实践中，可用非重极点的下列程序来求解。

◆MATLAB 程序

```
clear, clf, N=input('输入放大器级数 N=');
wn=1000; dt=1e-4; tf=0.01; t = 0: dt: tf;
y=zeros(N, length(t));                 % 输出初始化
for n=1: N
  p0=-linspace(.95, 1.05, n) * wn;     % 将 H(s)极点分散布置
  ay = poly([p0, 0]);     % 由 Y(s)的极点(比 H(s)多一个零极点)求分母系数
  by = prod(abs(p0));                  % 求 Y(s)的分子系数
  [r, p] = residue(by, ay);            % 求 Y(s)的留数极点
  for k = 1: n+1                       % 把各部分分式对应的时域分量相加
     y(n,: ) = y(n,: ) + r(k) * exp(p(k) * t);
  end
  figure(1), plot(t, y(n,: )); grid, hold on     % 绘制过渡过程曲线
  %下面这几条语句用来绘制波特图
  figure(2), bode(prod(abs(p0)), poly(p0)); hold on
  bh=by; ah= poly(p0);                 % 求 H(s)的分子分母系数
  w=logspace(2, 4);                    % 给出频率范围和分度
  H = polyval(bh, j * w)./polyval(ah, j * w); % 求 H(s)在各频点的值 H(jw)
  aH=unwrap(angle(H)) * 180/pi; % 求出以度为单位的连续相角
  fH=20 * log10(abs(H));               % 求出以分贝为单位的振幅
  figure(2),
  subplot(2, 1, 1), semilogx(w, fH), grid on, hold on       % 绘幅频图
  subplot(2, 1, 2), semilogx(w, aH), grid on, hold on       % 绘相频图
end, hold off
```

◆程序运行结果

运行此程序，设 N=4，可得 1 级到 4 级放大器的过渡过程如图 9-5，从中看出输出信号达到 0.6 处所需的时间约为单级时常数乘以级数。此程序在 N>4 时又会出现很大误差，读者可自己编写更好的程序。

为了画波特图，程序可按以下步骤进行：

（1）求频率特性。用多项式求值函数 polyval，并且用了元素群运算，输入频率数组作为自变量，一次就求出全部的频率特性。注意频率特性是复数，通常关心的是它们的振幅和相角。

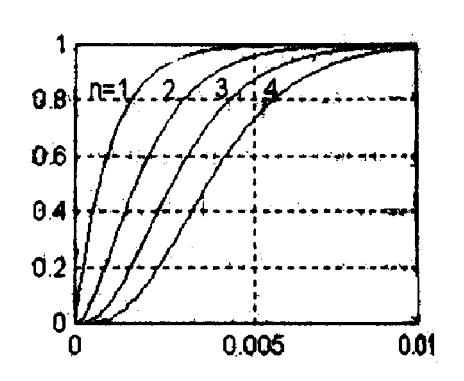
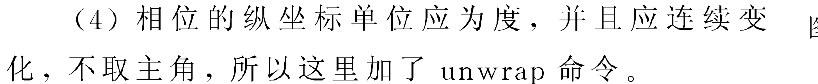


图 9－5　多级放大器的阶跃过渡过程

（2）振幅和相位特性横坐标都要用对数坐标并且上下对齐。

（3）振幅的纵坐标单位应化为分贝，这里用了 20 * log10(abs(H))。



（4）相位的纵坐标单位应为度，并且应连续变化，不取主角，所以这里加了 unwrap 命令。

在控制系统工具箱中有一个 bode 命令可以直接完成这些功能，但本书遵循的原则是不用工具箱，以便让读者知道工具箱是怎么编程的。当然读者也应知道，工具箱中的 bode 函数，远不止这几条语句，作为商用软件库中的一个正式函数，它必须考虑自动确定频率区间，自动检查输入有无错误等，程序要复杂得多。

图 9－6 绘出了多级放大器的频率特性。其幅频特性显示了低通的特点，随级数的增加，通带减小；相频特性说明，随级数的增加，负相移成比例地增加。

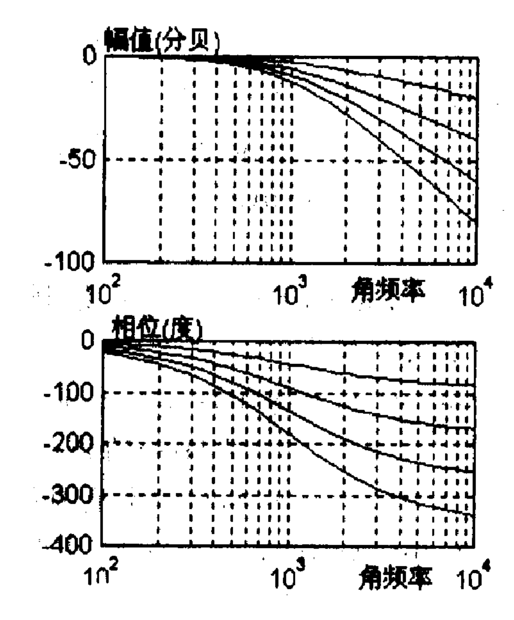


图 9－6　多级放大器的频率特性

【例 9－1－6】 方波可用相应频率的基波及其奇次谐波合成，这也是将它展开为正弦级数的出发点。现要求用 MATLAB 来演示这一结论。

解：

◆建模

一个以原点为奇对称中心的方波 y(t) 可以用奇次正弦波的叠加来逼近，即

$$y(t)=\sin t+\frac{1}{3}\sin 3t+\frac{1}{5}\sin 5t+\cdots+\frac{1}{(2k-1)}\sin(2k-1)t \quad (k=1,2,3,\cdots)$$

方波宽度为 π，周期为 2π，可以用 MATLAB 程序来检验这种逼近的程度和特征。

◆MATLAB 程序

```
t = 0:.1:10;                          % 设定一个时间数组有 101 个点
```

```
y = sin(t); plot(t, y), pause          % 频率为 w=1(f=1/2π)的正弦基波
y = sin(t) + sin(3 * t)/3; plot(t, y), pause    % 叠加三次谐波
% 用 1, 3, 5, 7, 9 次谐波叠加
y=sin(t)+sin(3 * t)/3+sin(5 * t)/5+sin(7 * t)/7+sin(9 * t)/9;
plot(t, y), pause
% 为了绘制三维曲面，要把各次波形数据存为一个三维数组，因此必须
% 重新定义 y，重新编程。由于打算求到 19 次谐波，把点取密一些
t = 0: .031: 3.14;
y = zeros(10, max(size(t))); x = zeros(size(t));
for k=1:2:19
x = x + sin(k * t)/k; y((k+1)/2,: ) = x;
end
% 将各波形叠合绘出
pause, plot(t, y(1: 9,: )),
% 将各波形绘成三维网格图，看出增加谐波阶次对方波逼近程度的影响
pause, mesh(y, k, t), pause
clc
```

◆程序运行结果

程序运行中将出现多幅画面，这里只给出最后得出的三维曲面图，如图 9－7 所示。取的阶次愈高，愈接近于方波，但总是消除不了边缘上的尖峰，这称为吉布斯效应。可以在命令窗内再键入命令

```
rotate3d
```

就可用鼠标拖动三维曲面旋转，从而更容易看地清吉布斯效应。

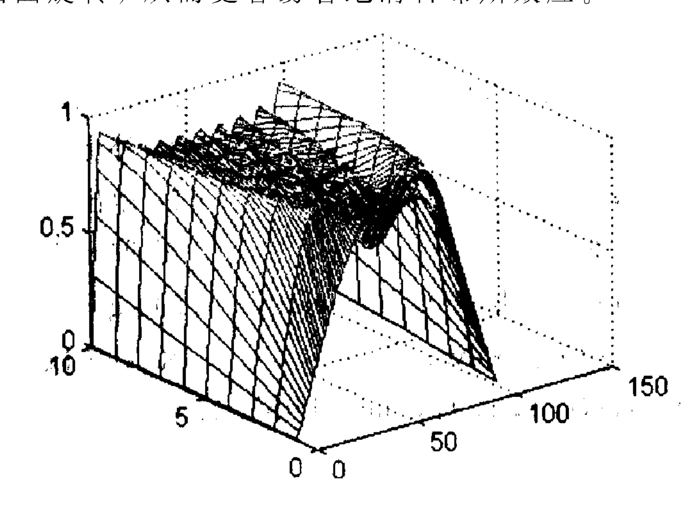


图 9－7 不同次数谐波叠合的三维曲面

【例 9－1－7】 设方波信号的宽度为 5 s，信号持续期为 10 s，试求其在 0～20 (1/s) 频段间的频谱特性。如只取 0～10(1/s)的频谱分量(相当于通过了一个低通滤波器)，求其输出波形。

解：

◆建模

设信号的时域波形为 f(t)，在 0～10 s 的区间外信号为 0，则其傅里叶变换为

$$F(j\omega)=\int_{-\infty}^{\infty}f(t)e^{-j\omega t}dt=\int_{0}^{10}f(t)e^{-j\omega t}dt$$

按 MATLAB 作数值计算的要求，必须把 t 分成 N 份，用相加来代替积分，对于任一给定的 ω，可写成

$$F(j\omega)=\sum_{i=1}^{N}f(t_n)e^{-j\omega t_n}\Delta t=[f(t_1),\cdots,f(t_N)]\cdot[e^{-j\omega t_1},\cdots,e^{j\omega t_N}]'\Delta t$$

这说明求和的问题可以用 f(t)行向量乘以 $e^{j\omega t}$ 列向量来实现。此处的 Δt 是 t 的增量，在程序中，将用 dt 来代替。

由于要求出一系列不同的 ω 处的 F 值都用同一公式，因此可以利用 MATLAB 中的元素群运算能力，把 ω 设成一个行数组，分别代入本公式左右端的 ω 中去，写成（程序中把 ω 写成 w）

$$F=f*\exp(-j*t'*w)\cdot\Delta t$$

其中，F 是与 w 等长的行向量，exp 中的 t′是列向量，w 是行向量，t′ * w 是一个矩阵，其行数与 t 相同，列数与 w 相同。这个式子就完成了傅里叶变换。类似地可以得到傅里叶逆变换表示式。由此得到下面的傅里叶变换程序。

◆MATLAB 程序

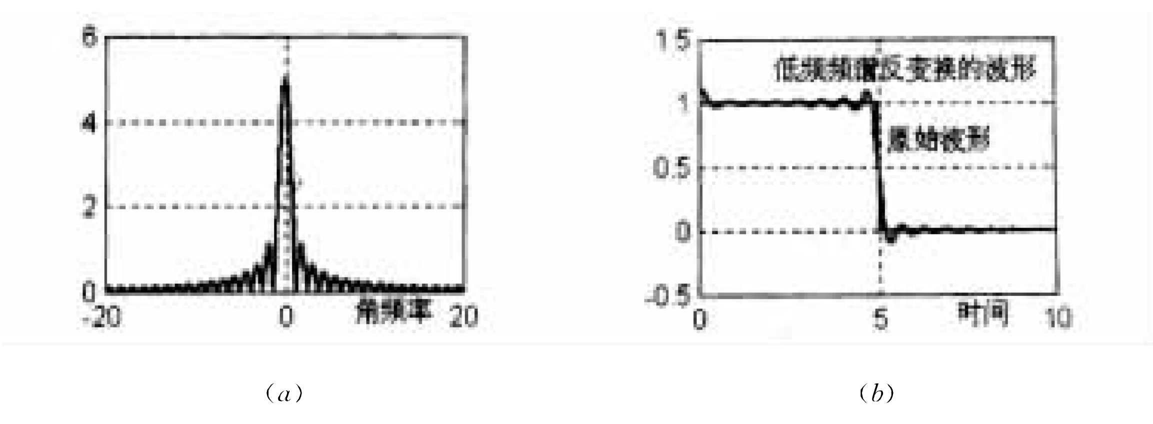
```
clear, tf=10; N=256;
t = linspace(0, tf, N);                          % 给出时间分割
w1 = linspace(eps, 20, N); dw =20/(N-1);
% dw=1/4/tf; w1 =[eps: dw: (N-1)/4/tf];          % 给出频率分割
f =[ones(1, N/2), zeros(1, N/2)];                % 给出信号(此处是方波)
F1 = f * exp(-j * t' * w1) * tf/(N-1);           % 求傅里叶变换
w = [-fliplr(w1), w1(2: N)];                     % 补上负频率
F = [fliplr(F1), F1(2: N)];                      % 补上负频率区的频谱
w2 = w(N/2: 3 * N/2);                            % 取出中段频率
F2 = F(N/2: 3 * N/2);                            % 取出中段频谱
subplot(1, 2, 1), plot(w, abs(F), 'linewidth', 1.5), grid
f1=F2 * exp(j * w2' * t)/pi * dw;                % 对中段频谱求傅里叶逆变换
subplot(1, 2, 2), plot(t, f, t, f1, 'linewidth', 1.5), grid
```

◆程序运行结果

执行这个程序的结果如图 9－8 所示，因为方波含有很丰富的高频分量，要充分恢复其原来波形需要很宽的频带，实践中不可能完全做到。

9.2 离散信号和系统

信号可以分为模拟信号和数字信号。模拟信号用 x(t)表示，其中变量 t 代表时间。离



(a)　　　　　　(b)

图 9－8　方波信号的频谱(a)和取|ω|<10 部分频谱的逆变换波形(b)

散信号用 x(n)表示，其中变量 n 为整数并代表时间的离散时刻，因此它也称为离散时间信号。离散信号是一个数字的序列，并可以表述为

$$x(n)=\{x(n)\}=\{\cdots,\ x(-1),\ \underset{\uparrow}{x(0)},\ x(1),\ \cdots\}$$

其中，向上的箭头表示在 n=0 处的取样。

在 MATLAB 中，我们可以用一个向量 x 来表示一个有限长度的序列。然而这样一个向量并没有包含基准采样位置的信息。因此，完全地表示 x(n)要用 x 和 n 两个向量。例如序列 $x(n)=\{2,1,-1,\underset{\uparrow}{5},1,4,3,7\}$(下面的箭头为第 0 个采样点)，在 MATLAB 中表示为

$$n=[-3,-2,-1,0,1,2,3,4],\ x=[2,1,-1,5,1,4,3,7]$$

当不需要采样位置信息或这个信息是多余的时候(例如该序列从 n=1 开始)，我们可以只用 x 向量来表示。由于内存有限，因此 MATLAB 无法表示无限序列。

【例 9－2－1】 编写 MATLAB 程序来产生下列基本脉冲序列。

(1) 单位脉冲序列：起点 n_0，终点 n_f，在 n_s 处有一单位脉冲($n_0 \leqslant n_s \leqslant n_f$)。

(2) 单位阶跃序列：起点 n_0，终点 n_f，在 n_s 前为 0，在 n_s 处及以后为 1($n_0 \leqslant n_s \leqslant n_f$)。

(3) 实数指数序列：$x_3=(0.9)^n$。

(4) 复数指数序列：$x_4=e^{(-0.2+0.3j)n}$。

解：

◆建模

这些基本序列的表达式比较简明，编写程序也不难。对单位脉冲序列，我们提供了直接赋值和逻辑关系两种方法，其中用逻辑关系的编法比较简洁，读者从中可看到 MATLAB 编程的灵活性和技巧性。通常用 stem 语句来绘制离散序列。

◆MATLAB 程序

```
clear, no=0; nf=10; ns=3;
n1=n0:nf; x1=[zeros(1,ns-n0),1,zeros(1,nf-ns)];
% n1 = n0:nf; x1=[(n1-ns)==0];          % 显然，用逻辑式是比较高明的方法
n2=n0:nf; x2=[zeros(1,ns-n0),ones(1,nf-ns+1)];
% 也有类似的用逻辑比较语句产生单位阶跃序列的方法，留给读者思考
n3 = n0:nf; x3=(0.9).^n3;               % 实数指数序列
n4 = n0: nf; x4=exp((-0.2+0.3j) * n3);  % 复数指数序列
```

```
subplot(2, 2, 1), stem(n1, x1);
subplot(2, 2, 2), stem(n2, x2);
subplot(2, 2, 3), stem(n3, x3);
subplot(4, 2, 6), stem(n4, real(x4));        % 注意 sunplot 的输入变元
subplot(4, 2, 8), stem(n4, imag(x4));
line([0, 10], [0, 0]),                        % 画横坐标
```

◆程序运行结果(如图 9－9 所示)

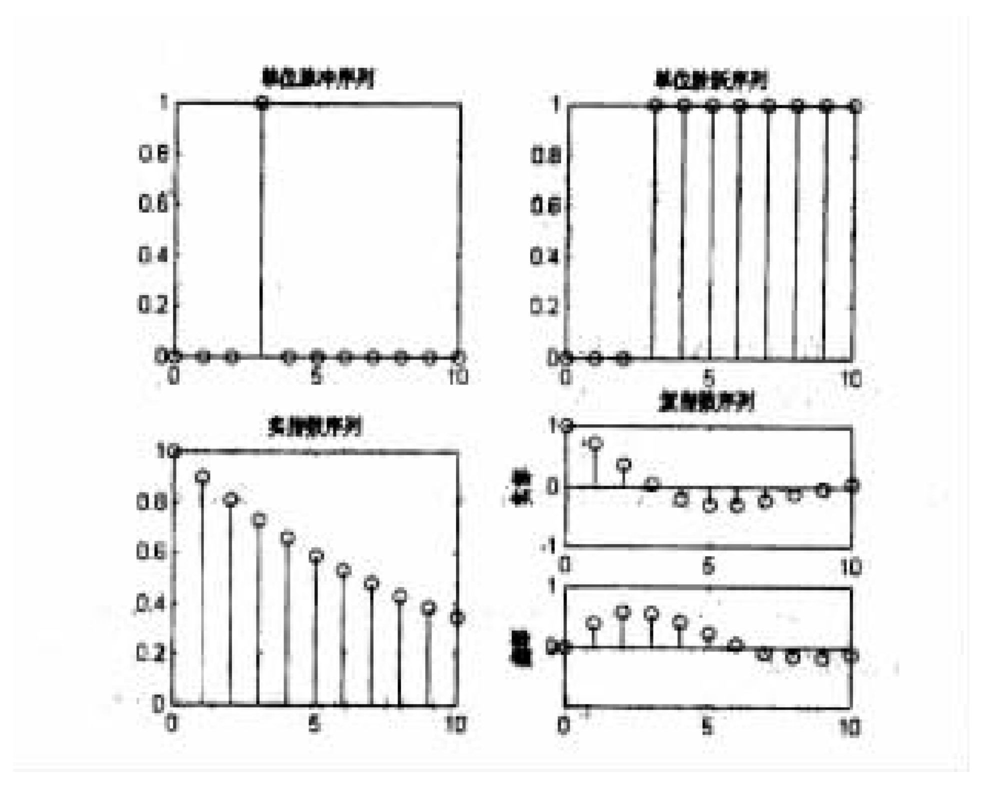


图 9－9 基本脉冲序列的波形

【例 9－2－2】 试编写计算线性时不变系统差分方程的通用程序。

解：

◆模型

线性时不变系统的特性可用差分方程描述为

$$a_1 y(n)+a_2 y(n-1)+\cdots+a_{na} y(n-n_a+1)=b_1 u(n)+b_1 u(n-1)+\cdots+b_{n_b} u(n-n_b+1)$$

因此其左端和右端的系数向量就完全描述了系统的特性。如果给定系统的输入，则系统的输出可表为(用 MATLAB 的表示法)

$$a(1)\cdot y(n)=b(1)*u(n)+b(2)*u(n-1)+\cdots+b(nb)*u(n-nb+1)$$
$$-a(2)*y(n-1)-\cdots-a(na)*y(n-na+1)$$

令 $us=[u(n),\cdots,u(n-nb)]$; $ys=[y(n-1),\cdots,y(n-na+1)]$

则上式可写成

$$a(1)*y(n)=b*us'-a(2:na)*ys'$$

这个方程是可以递推计算的。

这里遇到了一个困难，MATLAB 的变量下标只允许取正数，而这个公式需要知道 n=1 之前的 y 和 u。本例中的处理方法是另设两个变量 ym 和 um，它们相当于把 y 和 u 右

移 na－1 位，故 ym 和 um 的第 1 到 na－1 位相当于 y 和 u 在起点之前的初值。注意在程序中，随着计算点的右移，要随时生成相应于公式中的向量 us 和 ys。

◆MATLAB 程序

```
clear
a = input('差分方程左端的系数向量 a=[a(1), … a(na)]= ');
b = input('差分方程右端的系数向量 b=[b(1), … b(na)]= ');
u = input('输入信号序列 u= ');
na=length(a); nb=length(b); nu=length(u);
s=['起算点前', int2str(na-1), '点 ym 的值 =[ym(1), …, ym(na-1)]= '];
ym=zeros(1, na+nu-1);              % 先预设 ym 序列长 na+nu，并令初始化为 0
ym(1:na-1) = input(s);             % 给 ym 序列的前 na 个点赋予初值
um = [zeros(1, na-1), u];          % 建立 um 序列并赋予初值
for n=na:na+nu-1                   % 这个 n 以 ym 的起点为准
   ys = ym(n-1: -1:n-na+1); us = um(n-1:-1:n-nb); % 生成 us 及 ys
   ym(n) = (b * us'-a(2:na) * ys')/a(1);        % 差分方程递推求 ym
end
% 把 ym 时间坐标左移 na-1 位，求出 y
y = ym(na: na+nu-1);
stem(y), grid
line([0, 60], [0, 0])        % 给出起点和终点坐标，画横坐标轴
```

◆程序运行结果

执行此程序，输入

```
a=[1, -1, 0.9]; b=[2, 3]
u=cos(0.5 * [1: 59]);
ym=[0, 0];
```

得出的结果如图 9－10 所示。

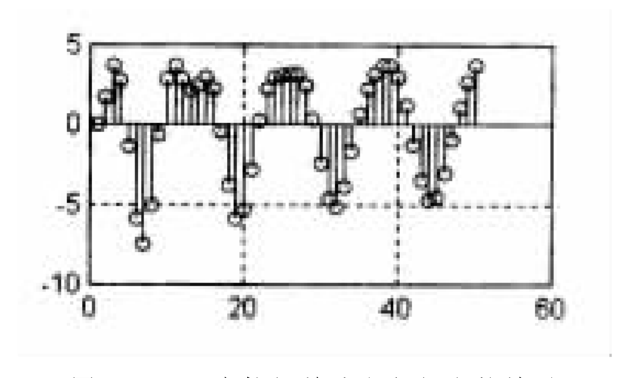


图 9－10　直接解差分方程得出的结果

【例 9－2－3】　离散傅里叶变换的计算。

解：

◆建模

一个时间序列 x(n)的离散时间傅里叶变换的定义为

$$X(e^{j\omega}) \triangleq F[x(n)] = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n}$$

如果序列的长度是有限的，可以把它看作是周期性无限序列中的一个周期，其长度为 N。对这个周期性序列可以用离散傅里叶变换（注意少了“时间”两字）进行研究，它的定义为

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j\frac{2\pi}{N}nk} = \sum_{n=0}^{N-1} x(n)W_N^{nk}$$

其中
$$W_N = \exp\left(-\frac{j2\pi}{N}\right)$$

用例 9－1－5 中的方法，引入矩阵乘法来实现求和运算。用元素群算法来求不同 k 时的 X，把 n 和 k 都设成 1×N 的行数组，令 nk＝n′＊k，它就成为 N×N 的方阵，因而 W_n^{nk} 也是N×N 方阵。由此得出离散傅里叶变换的算式

$$X = x * W_N^{nk}$$

MATLAB 只能处理有限长度的序列，因此，适合于计算离散傅里叶变换及其逆变换。

◆MATLAB 程序

设有限信号序列 xn(n)的长度为 Nx，则按定义，求其 N 点傅里叶变换 Xk(k)的程序为：

```
xn=input('x = '); Nx= length(xn); N=Nx       % 取 N 为 x 的长度
tic, n = [0: 1: N-1]; k = [0: 1: N-1];       % 设定 n 和 k 的行向量
WN = exp(-j * 2 * pi/N);                     % W_N 因子
nk = n' * k;                    % 产生一个含 nk 值的 N×N 维矩阵
WNnk = WN .^ nk;                             % 换算矩阵
Xk = xn * WNnk; toc          % DFT 系数向量，即离散傅里叶变换的结果
plot(abs(Xk)), grid                          % 绘幅频特性图
```

在 N 很大时，这个程序的运算速度比较低。程序中的 tic 和 toc 语句用来测试它们之间的程序运行时间。

实际上 MATLAB 已提供了快速离散傅里叶变换的函数 fft，可直接调用。其调用格式为

$$X = fft(x, N)$$

其中，x 是输入的时间序列，N 是傅里叶变换取的点数。若省略 N，则它自动把 x 的长度作为 N。当 N 取 2 的幂时，变换速度最快，所以要提高 fft 函数的运行速度，程序应编写如下：

```
xn=input('x =    '); Nx=length(xn)
% 取 N 为大于 Nx 而最接近于 Nx 的 2 的幂
N= pow2(nextpow2(Nx));
tic, X=fft(xn, N); toc
```

Nx<N，x 长度不足 N 的部分，程序会自动补 0。要注意 X 是一个长度为 N 的复数数组，可以分解出它的振幅和相位，分别绘图。

◆程序运行结果

令 n=1：700；

x=sin(0.1*n)+randn(1，700)；

然后依次在上述两个程序中输入这个 x，所得的 fft 的幅度特性是一样的，如图 9-11 所示。

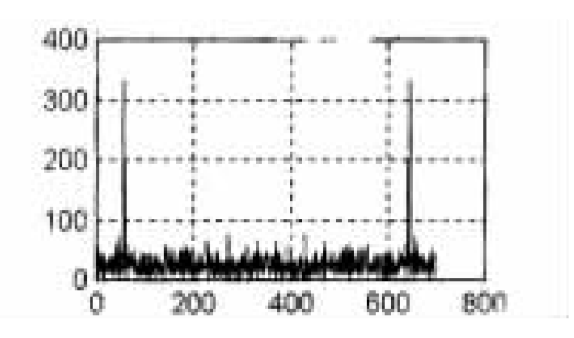


图 9-11　信号的 fft 的振幅频率特性

在作者的计算机上，前一程序的时间测试结果约为 8 s，后一程序为 0.05 s。

计算逆离散傅里叶变换的程序与此相仿，留待读者自行编写。

9.3　控制理论基础

【例 9-3-1】　求连续系统的传递函数表达式与零极点表达式之间的关系。

解：

◆建模

连续系统模型的典型表达式有传递函数法、零极增益法、极点留数法和状态空间法四种，这里讨论前三种。

1. 传递函数法

$$H(s) = f(s)/g(s)$$

对于单输入单输出系统，F(s)是 s 的 nf−1 次多项式，g(s)是 s 的 ng−1 次多项式，即

$$f(s)=f(1)s^{nf-1}+f(2)s^{nf-2}+\cdots+f(n_f-1)s+f(n_f) \qquad (1)$$

$$g(s)=g(1)s^{ng-1}+g(2)s^{ng-2}+\cdots+g(n_g-1)s+g(n_g) \qquad (2)$$

因此，知道分子系数矢量 f=[f(1)，f(2)，…，f(n_g)]和分母系数矢量 g=[g(1)，g(2)，…，g(ng)]，就决定了系统的模型。对物理可实现的系统，必有 $n_g \geqslant n_f$，系统的阶次为 $n=n_g-1$。

2. 零极增益法

对(1)式及(2)式进行因式分解，可得

$$H(s)=\frac{k(s-z(1))(s-z(2))\cdots(s-z(n_z))}{(s-p(1))(s-p(2))\cdots(s-p(n_p))} \qquad (3)$$

令 z=[z(1)，z(2)，…，z(n_z)]为系统的零点矢量，p=[p(1)，p(2)，…，p(n_p)]为系统的极点矢量，可以看出，零点的个数为 $n_z=n_f-1$，极点的个数为 $n_p=n_g-1=n$。对一切物理系统，有 $n \geqslant n_z$。系统的模型将由零极点矢量 z、p 及增益 k 所惟一地确定，故称为零极增益模型。

3．极点留数法

将零极增益模型(3)分解为部分分式，可得

$$H(s)=\frac{r(1)}{s-p(1)}+\frac{r(2)}{s-p(2)}+\cdots+\frac{r(n)}{s-p(n)}+h$$

其中，p=[p(1)，r(2)，…，p(n)]仍为极点矢量，r=[r(1)，r(2)，…，r(n)]为对应于各极点的留数矢量，因此 p、r、h 三者惟一地决定了系统的模型；h 是直通项，当分母阶次高于分子阶次时，h=0。前面已知，用极点留数模型易于得到脉冲过渡函数。

知道任一模型参数，用 MATLAB 很容易变换为另一种模型的参数。例如，给定 f 和 g，则

零极增益参数　z=roots(f)，p=roots(g)，k=f(1)/g(1)

极点留数参数　[r，p，h] = residue(f，g)

若知道零极增益参数 z、p、k，则传递函数的分子与分母为

f=poly(z) * k，g=poly(p)

若知道极点留数参数 r、p、h，则传递函数的分子与分母为

[f，g]=residue(r，p，h)

residue 函数具有双向变换的功能，变换的方向取决于输入变元的数目，输入变元为两个，则由 f、g 求 r、p、h，若输入变元为三个，则由 r、p、h 求 f、g。

由此，这三种模型之间的转换就全部解决了。还有一种状态空间模型，其变换也不难用 MATLAB 实现，这将在例 9 - 3 - 4 中介绍。

设系统的传递函数为

$$H(s)=\frac{8s^2+2s+1}{s^3+3.3s^2+9.9s+2.7}$$

求其零极增益表示式及其对应的脉冲过渡函数。

◆MATLAB 程序

```
f=[8，2，1]，g=[1.0000，3.3000，9.9000，2.7000]，
z=roots(f)，p=roots(g)，k=f(1)/g(1)，
[r，p，h] = residues(f，g)；
t=0：0.1：10；
x = r(1) * exp(p(1) * t) + r(2) * exp(p(2) * t) + r(3) *  exp(p(2) * t)；
plot(t，x)，
```

◆程序运行结果

执行程序的结果为

```
z = −0.1250 + 0.3307i，−0.1250 − 0.3307i
p = −1.5000 + 2.5981i，−1.5000 − 2.5981i，−0.3000
k = 8
```

过渡过程曲线如图 9 - 12 所示。

【例 9 - 3 - 2】　假设用传递函数模型，求两个环节串联、并联、反馈组成的新系统的参数。

解：

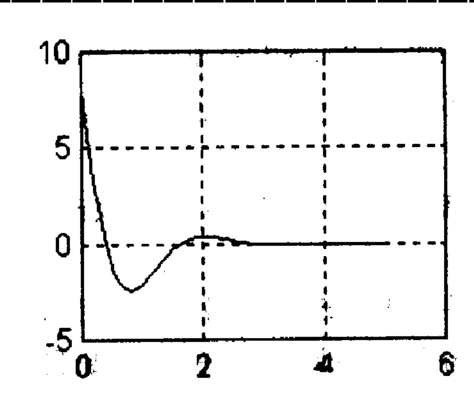


图 9－12　例 9－3－1 的过渡过程曲线

◆建模

传递函数模型。设 A、B 两个环节的传递函数分别为 $\frac{f_A(s)}{g_A(s)}$ 和 $\frac{f_B(s)}{g_B(s)}$，合成系统的传递函数为 $\frac{f(s)}{g(s)}$，则其公式如下：

串联：
$$H(s)=\frac{f(s)}{g(s)}=\frac{f_A(s)\cdot f_B(s)}{g_A(s)\cdot g_B(s)}$$

并联：
$$H(s)=\frac{f(s)}{g(s)}=\frac{f_A(s)}{g_A(s)}+\frac{f_B(s)}{g_B(s)}=\frac{f_A(s)\cdot g_B(s)+f_B(s)\cdot g_A*(s)}{g_A(s)g_B(s)}$$

反馈：(假定 B 在反馈通路上)

$$H(s)=\frac{f_A(s)g_B(s)}{f_A(s)f_B(s)+g_A(s)g_B(s)}$$

这是控制理论中的基本公式。现在来看如何用 MATLAB 来计算合成系统的系数 f 和 g。

◆MATLAB 程序

(1) 串联。对于串联的环节，需将两个环节分子和分母多项式分别相乘。在 MATLAB 中，多项式相乘可用 conv 命令来完成

```
f=conv(fA, fB); g=conv(gA, gB)
```

(2) 并联。对于并联的环节，需将其分子和分母多项式相加，通分以后仍然归结为多项式相乘和相加，因此也容易得到其在 MATLAB 中的表达式

```
f = conv(fA, gB)+conv(fB, gA);
g = conv(gA, gB);
```

(3) 反馈。

```
f = conv(fA, gB);
g = conv(fA, fB)+conv(gA, gB)
```

并联和反馈的表达式实际上往往无法运行，因为其中没有考虑到多项式相加时的限制条件，即两个多项式的长度必须相同才能相加。在长度不等的情况下，要把较短的那个前面补 0，因此简单地用"＋"号是不行的，要编一个子程序 polyadd.m，其内容为：

```
function y=polyadd(x1, x2)
n1=length(x1); n2= length(x2);
if n1>n2 x2=[zeros(1, n1-n2), x2];
elseif n1<n2 x1=[zeros(1, n2-n1), x1]; end
```

```
y = x1+x2;
```

原程序中 g = conv(fA, fB)+conv(gA, gB)应该改为 g=polyadd(conv(fA, fgB), conv(gB, gA)), 依此类推。

在 MATLAB 控制工具箱中，串联环节计算的函数为 series，并联环节计算的函数为 parallel，反馈环节计算的函数为 feedback，其后的变元数目可以为四个或八个。四个变元表明是传递函数法，八个变元表明是用状态空间法。这些函数会根据输入变元数目的不同自动调用相应的程序。

◆程序运行数字例

设 $f_A(s)=3s^2+2s+1$，$g_A(s)=5s^3+4s^2+7s+3$，$f_B=5s$，$g_B(s)=9s^2+8s+1$，求两者并联后的系统传递函数。

该数字例程序如下：

```
fA=[3, 2, 1]; gA=[5, 4, 7, 3]; fB=[5, 0]; gB=[9, 8, 1];
f = polyadd(conv(fA, gB), conv(fB, gA)), conv(gA, gB)
```

程序运行结果为

```
f = 52    62    63    25    1
g = 45    76    100    87    31    3
```

【例 9-3-3】 求线性定常系统用状态方程表示时的齐次解。

解：

◆建模

线性定常系统在输入信号为零时的状态方程表示式为

$$\dot{x}=Ax$$

其中 x 为 N×1 的列向量，而 A 为一个 N×N 的常系数方阵。此齐次方程的解为

$$x(t)=e^{At}x(0)=e^{A(t-t_0)}\cdot x(t_0)$$

其中，指数矩阵 $e^{A(t-t_0)}$ 是 N×N 阶的，它的另一个名字叫状态转移矩阵 $\Phi(t-t_0)$，因为它把 t_0 时的状态变量 $x(t_0)$变换为 t 时刻的状态变量 x(t)。在定常系统，$\Phi(t-t_0)=e^{A(t-t_0)}$。为了与 MATLAB 的符号衔接，后面将把 Φ 改为 F，初始条件 x(0)改为 x_0，它是 N×1 的列向量。从上式可以看出，求状态方程解的问题，很大程度归结为矩阵指数的计算，这有很多种方法，但通常都很繁。用手工计算，三阶以上就很难想象，MATLAB 在这方面有它特殊的优势。较简单的矩阵指数算法是直接用级数计算，即

$$e^{At}=1+At+\frac{1}{2!}A^2t^2+\frac{1}{3!}A^3t^3+\cdots+\frac{1}{k!}A^kt^k+\cdots$$

在 5.3 节中讨论过标量指数的计算问题，并给出了数例，但求矩阵指数时若考虑它的收敛性就不那么容易计算了，因此在这里可直接调用 MATLAB 的矩阵指数函数 expm。注意它与 exp 函数不同，多了一个 m，表示对矩阵整体求指数。其同类的函数还有 expm1、expm2、expm3，都是用来计算矩阵指数的，只是采用的方法不同而已。

矩阵指数计算比标量指数计算繁得多。如果 A 是 N×N 阶(矩阵指数只能用于方阵)，则 expm(A＊t)也是 N×N 阶(矩阵指数只能用于方阵)，这时如果要算一系列的 t 值所对应的 expm(A＊t)，就不可能像标量指数那样用元素群运算方法了，必须用 for 循环。不仅如此，如果 t 的长度为 Nt，则状态转移矩阵 F(t)=expm(At)将是一个 N×N×Nt 的三维

矩阵。

数字示例：设 A=[−2，1；−17，−4]，x0=[3；4]，求上述线性齐次方程系统的状态转移矩阵 F(t)及 x 的解。

◆MATLAB 程序　xx933

```
% 用多维矩阵求状态转移矩阵及齐次状态方程解
A=[-2，1；-17，-4]；                 % 输入状态方程系数矩阵
x0=[3；4]；                          % 输入初始条件
t=0：.02：3；Nt=length(t)；          % 设定自变量数组并确定其长度
F=zeros(2，2，Nt)；x=zeros(2，Nt)；% 状态转移矩阵 F 及状态变量初始化
for k=1：Nt                          % 对时间循环计算
   F(：，：，k)=expm(A*t(k))；        % 计算各时刻的状态转移矩阵 F
end
z=reshape(F，[4，Nt])；              % 把 F 变为二维矩阵以便绘图
plot(t，z)，grid，                   % plot 语句只接受二维变量
for k=1：Nt                          % 对时间循环，求各点状态变量
% 矩阵乘法只能用于二维，因此对每一时刻的 F，用 squeeze 函数缩去长度为 1 的第
% 三维，即把 F(：，：，k)看作 F(：，：)
   x(：，k)=squeeze(F(：，：，k))*x0；
end
figure，plot(t，x)，grid              % 画第二张图
```

◆程序运行结果

输出的数据此处予以省略，输出的曲线如图 9－13 所示，图上的标注都是用 gtext 命令完成的。在这个过程中，为了弄清各条曲线所代表的变量，必须查看各变量的数据。

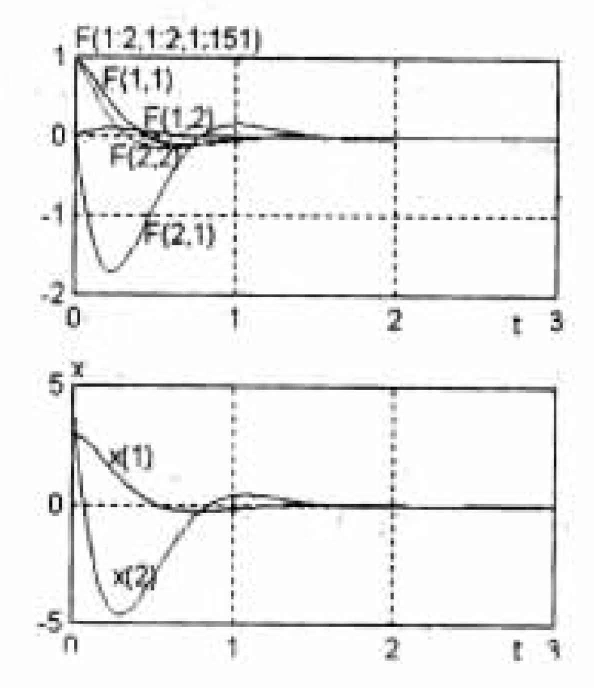


图 9－13　状态转移矩阵和状态变量的数值解曲线

【例 9－3－4】 已知连续系统的状态空间表达式，求传递函数表达式的系数向量。

解：

◆建模

控制系统的特点之一是多环节和多变量，因此，必须采用状态空间表示法。MATLAB 的矩阵运算优势在这方面更有作为，所以它最早也最广泛地用于控制理论中，除了有 control 工具箱外，还有其它几个工具箱也是专为新的控制理论开发的。本例只是给读者一个引子，在不用工具箱的情况下，初步介绍用于控制系统状态空间表示式的计算。

线性时不变系统的状态空间表示式如下：

$$\dot{x}=Ax+Bu$$
$$y=Cx+Du$$

如果系统是 n 阶的，输入有 n_u 个，输出有 n_y 个，则 A 为 $n\times n$ 阶，B 为 $n\times n_u$ 阶，C 为 $n_y\times n$ 阶，而 D 为 $n_y\times n_u$ 阶。对单输入单输出(SISO)系统，$n_y=n_u=1$。知道 A、B、C、D，就建立了系统模型。

n 阶系统的传递函数也可用其分子分母多项式的系数向量 f 和 g 描述。现在的问题是知道 A、B、C、D，如何求出 g、f 或惟之，知道 g、f，如何求出 A、B、C、D。因为状态空间表示法中有冗余参数，求 g，f 是惟一的，而反之则有无穷多解。

解：

◆建模

由 A、B、C、D 求 g，f。

对状态空间方程作拉普拉斯变换并求 H(s)＝Y(s)/U(s)，得

$$H(s)=C(sI-A)^{-1}B+D=\frac{f(s)}{g(s)} \tag{1}$$

$$(sI-A)^{-1}=\frac{\mathrm{adj}(sI-A)}{\det(sI-A)} \tag{2}$$

其中 adj(sI－A)为(sI－A)的随伴矩阵，即

$$\frac{C\ \mathrm{adj}(sI-A)B}{\det(sI-A)}+D=\frac{f(s)}{g(s)} \tag{3}$$

首先，使等式左右的分母相等。传递函数的分母 g(s)可分解为其诸特征根 lamda 的因式的乘积，即其多项式系数向量 g＝poly(lamda)，特征根可由 lamda＝eig(A)求得。于是有

$$g=\mathrm{poly}(\mathrm{eig}(A))=\det(sI-A) \tag{4}$$

令
$$(sI-A)^{-1}=\frac{\mathrm{adj}(sI-A)}{\det(sI-A)}=\frac{P(s)}{g(s)} \tag{5}$$

$$P(s)=\mathrm{adj}(sI-A)=P_1s^{n-1}+P_2s^{n-2}+\cdots+P_{n-1}s+P_n \tag{6}$$

为按 s 的降幂排列 $n\times n$ 阶系数方阵多项式，s 的最高次数为(n－1)，即它比 det(sI－A)低一阶，由此得

$$sIP(s)-AP(s)=g(s) \tag{7}$$

使等式左右 s 同次项的系数相等，可得以下的递推方程：

s^n： $P_1=I$

s^{n-1}： $P_2=AP_1+g_2I$

…　　(8)

s^{n-k}:　　$P_{k+1} = AP_k + g_{k+1}I$

…

P 本身是一个多项式方阵，它的不同下标表示 n 个系数方阵，这就需要三维的矩阵表示方法。MATLAB 5.x 提供了这种方法。

求出 P(:,:,i)后，即可由(3)式求得

$$f = C \cdot P(s) \cdot B + D \cdot g \tag{9}$$

◆MATLAB 程序

```
clear,
A=input('A= '), B=input('B= '), C=input('C= '), D=input('D= ')
g=poly(eig(A)); n=length(A);
P(:,:,1)=eye(n); f(1)=D*g(1);                 % 公式(8)的第一式
f(2)=C*P(:,:,1)*B+D*g(2);                     % 公式(9)的第一式
for i=2:n                                     % 循环递推
   P(:,:,i)=A*P(:,:,i-1)+g(i)*eye(n);         % 公式(8)
   f(i+1)=C*P(:,:,i)*B+D*g(i+1);              % 公式(9)
end, f, g
```

在这个程序中我们使用了三维矩阵 P(:,:,i)，这种多维矩阵是在 MATLAB 5.x 中新扩展的，所以这个程序只能在 MATLAB 5 的环境下才能运行。

◆程序运行结果

任意给定

```
A = 0.9501    0.8913    0.8214    0.9218
    0.2311    0.7621    0.4447    0.7382
    0.6068    0.4565    0.6154    0.1763
    0.4860    0.0185    0.7919    0.4057
B = 0.9355
    0.9169
    0.4103
    0.8936
C = 0.0579    0.3529    0.8132    0.0099
D = 0
```

运行上述程序可得

```
f =        0     0.7201   -0.0356    0.1562   -0.1142
g =   1.0000    -2.7334    1.2135   -0.6543    0.1155
```

在控制系统工具箱中，状态空间模型转换为传递函数模型的专门函数是 ss2tf。也有四种模型之间互相转换的函数，其基本的方法类似。在控制系统工具箱中可以直接调用这些函数。

9.4 偏微分方程数值解

【例 9－4－1】 二维电位函数的数值解法。

如图 9－14 所示，横截面为矩形的无限长槽由三块接地导体板构成，槽的盖板接以直流电压 U_0，求矩形槽中的电位分布。

解：

◆建模

[解法一] 分离变量法

在直角坐标系中，矩形槽中的电位分布 $\Phi(x,y)$满足二维拉普拉斯方程

$$\nabla^2\Phi=\frac{\partial^2\Phi}{\partial x^2}+\frac{\partial^2\Phi}{\partial y^2}=0$$

其边界条件按题意给定为

$$\Phi(x,y)\big|_{x=0}=0,\ \Phi(x,y)\big|_{x=a}=0$$

$$\Phi(x,y)\big|_{y=0}=0,\ \Phi(x,y)\big|_{y=b}=U_0$$

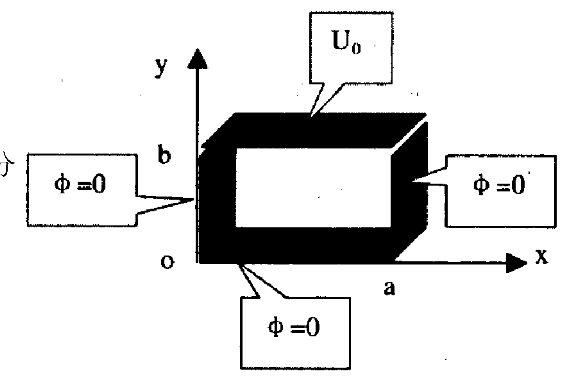


图 9－14 无限长槽的几何图形

采用分离变量求得电位分布严格解为

$$\Phi(x,y)=\frac{4U_0}{\pi}\sum_{n=1,3,5}^{\infty}\frac{1}{n\cdot \mathrm{sh}\frac{n\pi}{a}b}\mathrm{sh}\frac{n\pi}{a}y\cdot\sin\frac{n\pi}{a}x$$

[解法二] 有限差分法

有限差分法是一种数值解法。这种方法是将场域进行网格划分，并用有限差分方程近似地替代偏微分方程，然后求解差分方程，得到电位函数的数值解。

首先，导出差分方程。设在一个边界为 L 的二维矩形区域 D 内，电位函数 $\Phi(x,y)$满足二维拉普拉斯方程

$$\nabla^2\Phi=\frac{\partial^2\Phi}{\partial x^2}+\frac{\partial^2\Phi}{\partial y^2}=0 \qquad (1)$$

在边界 L 上各点的电位值已给定，即

$$\Phi(x,y)\big|_L=f(x,y) \qquad (2)$$

为了采用差分法求解 D 内的电位函数，作平行于坐标轴的两组平行线

$$x_i=x_0+ih,\quad y_j=y_0+jh \qquad (i,j=1,2,3,\cdots) \qquad (3)$$

将区域 D 划分为许多正方形网格，如图 9－15 所示。网格的边长 h 称为步长，两组平行线的交点称为网格的节点。

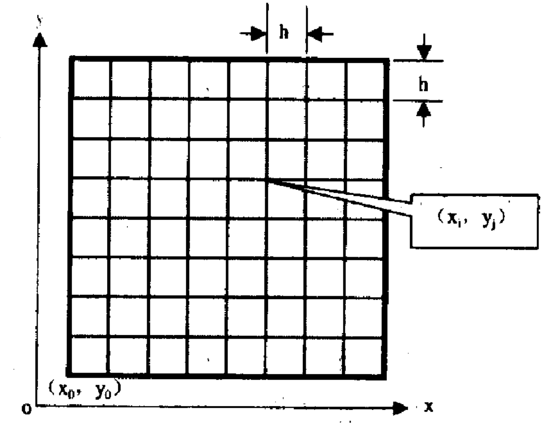


图 9－15 区域网格划分

我们用 $\Phi_{i,j}$表示节点(x_i,y_j)处的电位值。下面导出节点上的电位值所满足的差分方程。利用二元函数的泰勒展开式，可将与节点(x_i,y_j)直接相邻的节点上的电位函数值表示为：

$$\Phi_{i-1,j}=\Phi(x_i-h,\ y_j)=\Phi_{i,j}-h\left(\frac{\partial\Phi}{\partial x}\right)_{i,j}+\frac{h^2}{2}\left(\frac{\partial^2\Phi}{\partial x^2}\right)_{i,j}-O(h^3) \tag{4}$$

$$\Phi_{i+1,j}=\Phi(x_i+h,\ y_j)=\Phi_{i,j}+h\left(\frac{\partial\Phi}{\partial x}\right)_{i,j}+\frac{h^2}{2}\left(\frac{\partial^2\Phi}{\partial x^2}\right)_{i,j}+O(h^3) \tag{5}$$

$$\Phi_{i,j-1}=\Phi(x_i,\ y_j-h)=\Phi_{i,j}-h\left(\frac{\partial\Phi}{\partial y}\right)_{i,j}+\frac{h^2}{2}\left(\frac{\partial^2\Phi}{\partial y^2}\right)_{i,j}-O(h^3) \tag{6}$$

$$\Phi_{i,j+1}=\Phi(x_i,\ y_j+h)=\Phi_{i,j}-h\left(\frac{\partial\Phi}{\partial y}\right)_{i,j}+\frac{h^2}{2}\left(\frac{\partial^2\Phi}{\partial y^2}\right)_{i,j}+O(h^3) \tag{7}$$

其中，$O(h^3)$ 表示 h^3 量级的微量。将式（4）和式（5）相加，并忽略 $O(h^3)$ 项以及更高阶项可得

$$\left(\frac{\partial^2\Phi}{\partial x^2}\right)_{i,j}\approx\frac{1}{h^2}(\Phi_{i-1,j}-2\Phi_{i,j}+\Phi_{i+1,j}) \tag{8}$$

上式是 x 方向的二阶偏导数 $\left(\frac{\partial^2\Phi}{\partial x^2}\right)_{i,j}$ 的二阶差分表达式。同理，由式(6)和式(7)，可得到 y 方向的二阶偏导数 $\left(\frac{\partial^2\Phi}{\partial y^2}\right)_{i,j}$ 的二阶差分表达式为

$$\left(\frac{\partial^2\Phi}{\partial y^2}\right)_{i,j}\approx\frac{1}{h^2}(\Phi_{i,j-1}-2\Phi_{i,j}+\Phi_{i,j-1}) \tag{9}$$

将式(8)和(9)代入式(1)可得到节点$(x_i,\ y_j)$处的有限差分方程

$$\Phi_{i,j}=\frac{1}{4}(\Phi_{i-1,j}+\Phi_{i+1,j}+\Phi_{i,j-1}+\Phi_{i,j-1}) \tag{10}$$

这样，在节点$(x_i,\ y_j)$处的电位 Φ 满足的拉普拉斯方程近似地被一个差分方程所代替。由式(10)可见，用差分方程代替拉普拉斯方程后，原来的二阶偏导数运算转化为代数运算，节点$(x_i,\ y_j)$的电位值 $\Phi_{i,j}$ 仅由周围四个相邻节点的电位值决定，这就使问题的求解得以简化。

其次，求解差分方程(10)。由于对场域 D 内的每一个节点都有一个差分方程，因此得到的是一个差分方程组，这个方程组所包含方程的个数就等于场域 D 内的节点数。对实际问题，为了使所得到的解具有足够的精确度，节点数通常取的很多，因此差分方程组的阶数往往很高，需要使用计算机来求解这种高阶线形方程组。求解差分方程组最常用的方法是同步迭代法。同步迭代法是最简单的迭代方式。首先，任意给定区域 D 内每一个节点上的数值作为电位的零次近似值 $\Phi^{(0)}_{(i,j)}$，然后把这组值代入式(10)的右端得到

$$\Phi^{(1)}_{i,j}=\frac{1}{4}[\Phi^{(0)}_{i,j-1}+\Phi^{(0)}_{i,j+1}+\Phi^{(0)}_{i-1,j}+\Phi^{(0)}_{i+1,j}]$$

将 $\Phi^{(1)}_{i,j}$ 作为电位的一次近似值。在上式右端的四个值中若涉及到边界节点上的值时，均用相应的已知值 $f(x_i,\ y_j)$代入。再将 $\Phi^{(1)}_{i,j}$ 代入式(10)的右端，又可得到电位的二次近似值 $\Phi^{(2)}_{i,j}$。一般来说，在得到电位的第 k 次近似值 $\Phi^{(k)}_{i,j}$ 后，由公式

$$\Phi^{(k+1)}_{i,j}=\frac{1}{4}[\Phi^{(k)}_{i,j-1}+\Phi^{(k)}_{i,j+1}+\Phi^{(k)}_{i-1,j}+\Phi^{(k)}_{i+1,j}] \tag{11}$$

得到电位的第 k+1 次近似值。照这样进行下去，直到相邻两次的迭代解 $\Phi^{(k)}_{i,j}$ 与 $\Phi^{(k+1)}_{i,j}$ 间的误差不超过允许范围时，就可结束迭代过程。

下面取 a=16 ，b=9，U_0=100 V ，采用式(11)计算图 9－14 所示的横截面为矩形的

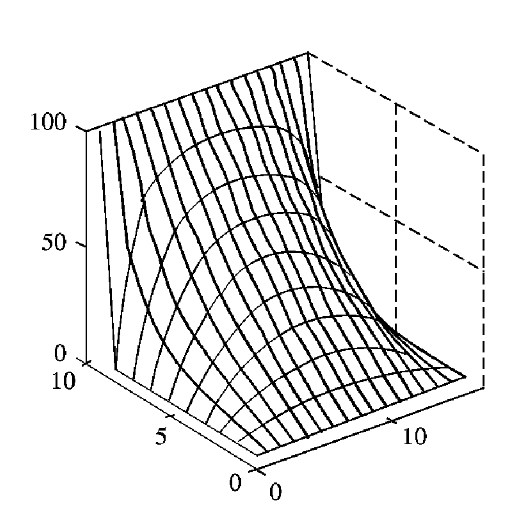
无限长加盖槽的电位分布。应用 MATLAB 语言的计算程序如下，其中 Φ 用 v 来表示。

◆MATLAB 程序

```
% 横截面为矩形的无限长加盖导体槽的电位
h=4; v0=100; hx=4*h+1; hy=3*h-2;
% 设定电位零次近似值 v1，盖上为 v0，其余为 0
v1=zeros(hy, hx); v1(hy,:)=ones(1, hx)*v0;
% 置初值，设离开边缘部分电位均为 v0/2
v1(2:hy-1, 2:hx-1)=ones(hy-2, hx-2)*v0/2; %
v2=zeros(hy, hx);                          % 初始化结果变量 v2
% ———先由 v1 迭代运算求 v2———————————————
for i=1:hy
    for j=1:hx
      if i==hy v2(i, j)=v0;                % 盖上电位为 v0
       elseif i==1|j==1|j==hx v2(i, j)=0;  % 其余三边电位为 0
       else v2(i, j)=(v1(i, j-1)+v1(i, j+1)+v1(i-1, j)+v1(i+1, j))/4;
            % 拉普拉斯方程
      end
    end
end
% ———把 v2 代替 v1，再由 v1 迭代运算求 v2，进入循环迭代—————
for k=1:500                          % 迭代次数 k=500
 v1=v2;
 for i=1:hy
   for j=1:hx
    if i==hy v2(i, j)=v0;            % 盖上电位为 v0
    elseif i==1|j==1|j==hx v2(i, j)=0;   % 其余三边电位为 0
    else v2(i,j)=(v1(i,j-1)+v1(i,j+1)+v1(i-1,j)+v1(i+1,j))/4;
        % 拉普拉斯方程
    end
   end
  end
end
subplot(1, 2, 1), mesh(v2)          % 画三维曲面图
axis([0, 17, 0, 10, 0, 100])
subplot(1, 2, 2), contour(v2)       % 画等电位线图
```

◆程序运行结果

数据结果此处从略。槽截面上的电位分布曲面如图 9－16 所示，底面表示槽截面，z 轴表示电位的大小，在 hy=10 处，电位为 100，其余三边电位为 0。



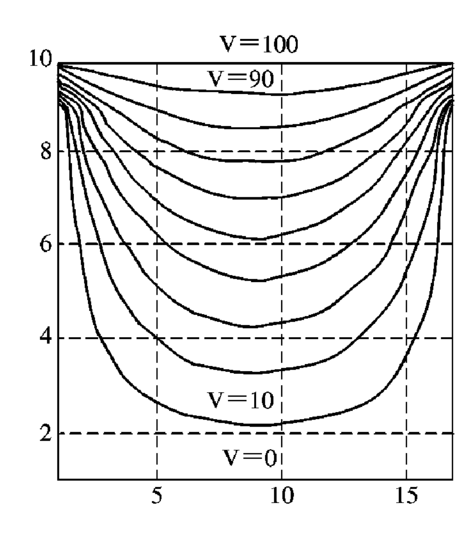


图 9-16　槽截面上的电位分布曲面(左)及等位线图(右)

在上述程序中，两套双 for 循环语句几乎是完全重复的。如果巧妙地运用 while 循环，就可以把他们合成一套程序，可节约十几行语句。我们把它命名为 ex941r，放在程序集中供读者参考。

第 10 章 MATLAB 工具箱简介

本书前九章介绍的是 MATLAB 的基本部分，它处在 MATLAB toolbox matlab 的子目录中。在比较完整的 MATLAB 专业版的工具箱(toolbox)子目录内实际上还有 20 多个其它子目录。一般所谓的工具箱，就是指 toolbox matlab 之外的子目录，因为它们需要单独选择购买，而子目录 matlab 则是任何版本的 MATLAB 都不可少的。这些工具箱可以分为两类：

(1) 有通用性的工具箱，它具有独立的开发环境和基础，不是 MATLAB 基本部分的简单扩展。因此它也具有基础意义。属于这类的有两个，即符号运算(Symbolic)和框图仿真(Simulink)工具箱。Symbolic 是利用 MATLAB 的界面，调用 MAPLE 软件的工具，MAPLE 是加拿大 Waterloo 大学开发的公式推理软件，在国际上开始最早，也是最为成功的。从 20 世纪 90 年代初 MATLAB 4 版本开始，Mathworks 公司就与 Maple 合作，推出 Symbolic。到 MATLAB 5，它更为完善。Simulink 是用框图来代替算式的 MATLAB 框图界面工具，虽然它全是 Mathworks 公司开发，而且是以 MATLAB 的基本部分为基础的，但它也有一套独立的程序结构和方法。

(2) 专用工具箱，它的绝大部分是以 MATLAB 基本部分为基础编写的子程序集，目的是解决特定的学科领域或应用领域的问题。这些工具箱中的程序语句，没有一条会超出本书的范围，有几个专用工具箱还必须用 Simulink。从商业角度考虑，这类工具箱之间没有较多的相互依赖关系，各个工具箱大多数可以独立选购，只有其基本部分是必须购买的。

10.1 符号数学(Symbolic Math)工具箱简介[14]

顾名思义，符号数学是以符号(如 a，b，c，x，y，z)为对象的数学，以区别于以数字为对象的 MATLAB 基本部分。在大学教育中，符号数学是每门课都用到的，因此 Mathworks 公司专门提供给大学生的版本(Student Edition of MATLAB)中就包括了这个工具箱。国外用 MATLAB 教授数学的教科书中，大概有 15%～20%的例题和习题会用到这个工具箱。本书没有把这个工具箱用到各课程中去，主要考虑到以下两点：

(1) 在大学教学中，推理是一个基本功，而且在大多数大学课程中，也没有太复杂的推理。一般说，把数值计算交给计算机去计算，绝大多数老师还是能接受的，如果把推理也交给计算机，对教学是否有利，这是一个有争议的问题。笔者认为，符号数学工具箱还不宜在本科低年级使用。

(2) Symbolic 工具箱对计算机硬件(包括外存，内存，时钟频率等)的要求比较高，在

大部分本科学生的机房内，难以达到要求。

1. Symbolic 工具箱的主要功能

(1) 用符号定义各种数学运算和函数(syms，symop)等。

(2) 对这些函数式进行代数和三角运算，包括因式分解(factor)、展开(expand)，变量置换(subs)和复合函数(compose) 等。

(3) 微分和积分运算(diff，int)等。

(4) 函数的整理和化简(combine，simplify，simple)等。

(5) 可变精度的运算，可以设置任意多的有效计算位数进行计算(vpa，digits) 等。

(6) 解方程，包括单变量的代数方程、多变量非线性的联立代数方程(solve)、单变量微分方程和多变量联立微分方程(dsolve)等。

(7) 线性代数和矩阵运算(determ，linsolve)等。

(8) 变换，包括拉普拉斯变换(laplace)、傅里叶变换(fourier)和 Z 变换(ztrans)等。

2. 符号数学式的基本表示方法

符号数学是对字符串进行的，在 MATLAB 中，如果键入

f=3 * x^2+5 * x+2 或 y=sin(x)

系统会指出：变量 x 无定义，因为它要求 x 必须是一个数。MATLAB 也可以接受形式为

f='3 * x^2+5 * x+2' 或 y='sin(x)'

的语句，这时，f 和 y 都是一个字符串，但它没有任何含义，因为它对字符串中的内容不作任何分析。

Symbolic 工具箱必须要能分析字符串的含义，为此首先要对符号变量作出定义，用语句

x = sym('x')

就定义了 x 是一个字符(串)变量，此后键入的算式 f=3 * x^2+5 * x+2 或 y=sin(x)就具有了符号函数的意义，不必再用引号来表明它为字符串，f 和 y 也自然成为字符(串)变量。

如果一个数学符号表示式中有多个符号，如

z=a * t^2+b * t+c

可以用多个符号变量定义语句放在此表示式的前面

syms a b c t

注意只需对算式的右边符号变量作定义。

在做符号运算时，例如求上述代数方程的根，就得知道哪个(些)是变量，哪个(些)是系数，这时可在语句中指定，例如 r=solve('z=0', t)就表示解方程 z=0，得出解 t。如果不加说明，键入 r=solve('z=0')，系统自动把 26 个字母排序中离 x 最近的那个(些)符号当作变量来求解。在本例中，自然会选到 t。

下面以表格方式列出一些算例，使读者对本工具箱的功能有一个大体的印象，如表 10－1所示。在表中，假定符号变量的定义已经给出，系统中已安装了符号数学工具箱，则键入第二列的符号式就会得到第三列的结果。

表 10－1 符号数学工具箱的功能示例

运算式	符号式	系统响应
符号函数赋值	r = x^2 + y^2	r =x^2+y^2
符号函数赋值	theta = atan(y/x)	theta = atan(y/x)
符号函数赋值	e = exp(i * pi * t)	e = exp(i * pi * t)
三角函数式化简	f = cos(x)^2 + sin(x)^2	f =cos(x)^2+sin(x)^2
	f1 = simple(f)	f1 =1
微分	diff(x^3)	ans =3 * x^2
积分	int(x^3)	ans =1/4 * x^4
	int(exp(－t^2))	ans =1/2 * pi^(1/2) * erf(t)
矩阵按元素群积分	I=[int(x^a), int(a^x), int(x^a, a), int(a^x, a)]	I=[x^(a+1)/(a+1), 1/log(a) * a^x 1/log(x) * x^a, a^(x+1)/(x+1)]
解二次代数方程	x=solve('a * x^2 + b * x + c = 0'); x	x = [1/2/a * (－b+(b^2－4 * a * c)^(1/2))] [1/2/a * (－b－(b^2－4 * a * c)^(1/2))]
解联立代数方程	[u, v] = solve('a * u^2 + v^2 = 0', 'u － v = 1')	u = [1/2/(a+1) * (－2 * a+2 * (－a)^(1/2))+1] [1/2/(a+1) * (－2 * a－2 * (－a)^(1/2))+1] v = [1/2/(a+1) * (－2 * a+2 * (－a)^(1/2))] [1/2/(a+1) * (－2 * a－2 * (－a)^(1/2))]
以 28 位有效数解联立超越方程	digits(28) [x, y]=solve('sin(x+y)－exp(x) * y = 0', 'x^2－y = 2')	x=－6.017327250059306564109729712 y=34.20822723430629650864621443
求一阶微分方程通解	y = dsolve('Dy = －a * y')	y =exp(－a * t) * C1(含任意常数)
给出初始条件求微分方程特解	y= dsolve('Dy = －a * y', 'y(0) = 1')	y =exp(－a * t)
求二阶微分方程特解，D2 表示二阶导数	% 给出两个初始(边界)条件 y= dsolve('D2y = －a^2 * y', 'y(0) = 1, Dy(pi/a) = 0')	y = cos(a * t)
求二阶微分方程特解	y = dsolve('(Dy)^2 + y^2 = 1', 'y(0) = 0')	y = [sin(t)] [－sin(t)] (有两个解)
拉普拉斯变换	f=exp(－a * t) * cos(w * t)	f =exp(－a * t) * cos(w * t)
	F=laplace(f)	F =exp(－a * t) * s/(s^2+t^2)
	pretty(F) 此命令用来改善公式可读性	$\frac{\exp(-at)s}{s^2+t^2}$

为了节省篇幅，这里尽量选简单的推导式，实际上还可以推导很繁的式子。目前 MATLAB 符号数学推理的限度是：表示式的长度不超过 1400 个字符。

在一般意义下，使用 MATLAB 公式推导是很方便的。只要不给自变量赋以数值，而代之以

　　syms　自变量 1　自变量 2　自变量 3　…

以后的编程和普通 MATLAB 程序完全相同。其执行的结果自然是表达式而不是数值解。如果要做进一步的工作，例如化简、代换、代入数值、解联立方程等，那就需要对这个工具箱有较完整的了解。

【例 10－1－1】 给定如下联立方程：

$$3x_1+5x_2-8=0$$

$$2x_1-7x_2+4=0$$

求 x_1，x_2。若将第二式改为 $2x_1-7x_2+4u=0$，求 x_1，x_2。

解：第一问有两种解法。

（解法 1）数值方法：将方程组经过移项，写成标准的矩阵方程 AX=B，其中

$$A=\begin{bmatrix}3 & 5\\2 & -7\end{bmatrix},\quad B=\begin{bmatrix}8\\-4\end{bmatrix}$$

用 MATLAB 解出

X=A　B

程序为

A=[3,5；2，－7]；B=[8；－4]；X=A　B

运行此程序得到 X=[1.1613；0.9032]，这两个数就是待求的 X=[x1；x2]。

（解法 2）符号数学方法：将方程组用字符串写成符号方程，字符串名为 s1 和 s2。然后用代数方程求解命令：

```
syms x1x2
s1='3*x1+5*x2-8=0'
s2='2*x1-7*x2+4=0'
[x1，x2]=solve(s1，s2)
```

执行这几条程序所得的结果为

```
x1=36/31，x2=28/31
```

不难看出，这两种解法所得的结果是相同的。

对于第二问，第一种解法就无能为力了。只能用解法 2。把上述程序 1，3 两条语句改为

```
syms x1 x2 u；s2='2*x1-7*x2+4u=0'
```

运行后得到

```
x1=56/31-20/31*u，x2=16/31+12/31*u
```

可以看出，符号运算的特点之一是不求出除法的数值结果，以避免运算误差的产生和积累。其次，它可以不必移项成规范形式。比如把 s2 改为 s2='2*x1=7*x2-4u'，解出的结果是相同的。

10.2　Simulink 工具箱简介

框图是方程的等价物，即使是普通的微分方程，也可以用框图来描述。在工程设计计算中，人们往往喜欢用框图来表示复杂的系统，因为它形象地把各个环节区分开来，并且便于研究各个环节的参数对系统的影响。Simulink 工具箱就提供了这样的方法。

以一个最简单的弹簧－质量系统为例，其结构如图 10－1 左下角所示。弹簧的左端是输入 x_1，弹簧的右端是一质量 M，弹簧刚度系数为 K，设质量 M 对于初始平衡位置的位移为 x_2，它与地面间的阻尼力与 M 的速度成正比，方向相反，阻尼系数为 C。于是 M 受弹簧和阻尼两个力，其运动方程可写成

$$M\ddot{x}_2 = K(x_1 - x_2) - C\dot{x}_2$$

这个方程可以用图 10－1 中带两个反馈的框图部分来描述。它们分别表示由于速度 $\dot{x}_2$ 和位置 x_2 的变化，使作用与 M 上的力 $K(x_1 - x_2)$和$-C\dot{x}_2$ 发生变化。框图的外围还有几个辅助的方框，左面两个是生成输入信号的信号源和滤波器，右边的两个是多路器和示波器，用以观察仿真过程中多个框图输出的波形，上方是动画生成器，仿真时会产生左下角的实体运动动画。

所有这些框图单元都存在 Simulink 的库中，可以将它们调出并以有向线段连接。这个工作可用鼠标很方便地完成，只要双击环节的框，就会出现对它设置参数的窗口，图10－1 左上角的视窗就是双击增益环节 K/M 后自动弹出的，可以用键盘修改 K/M 值。把环节参

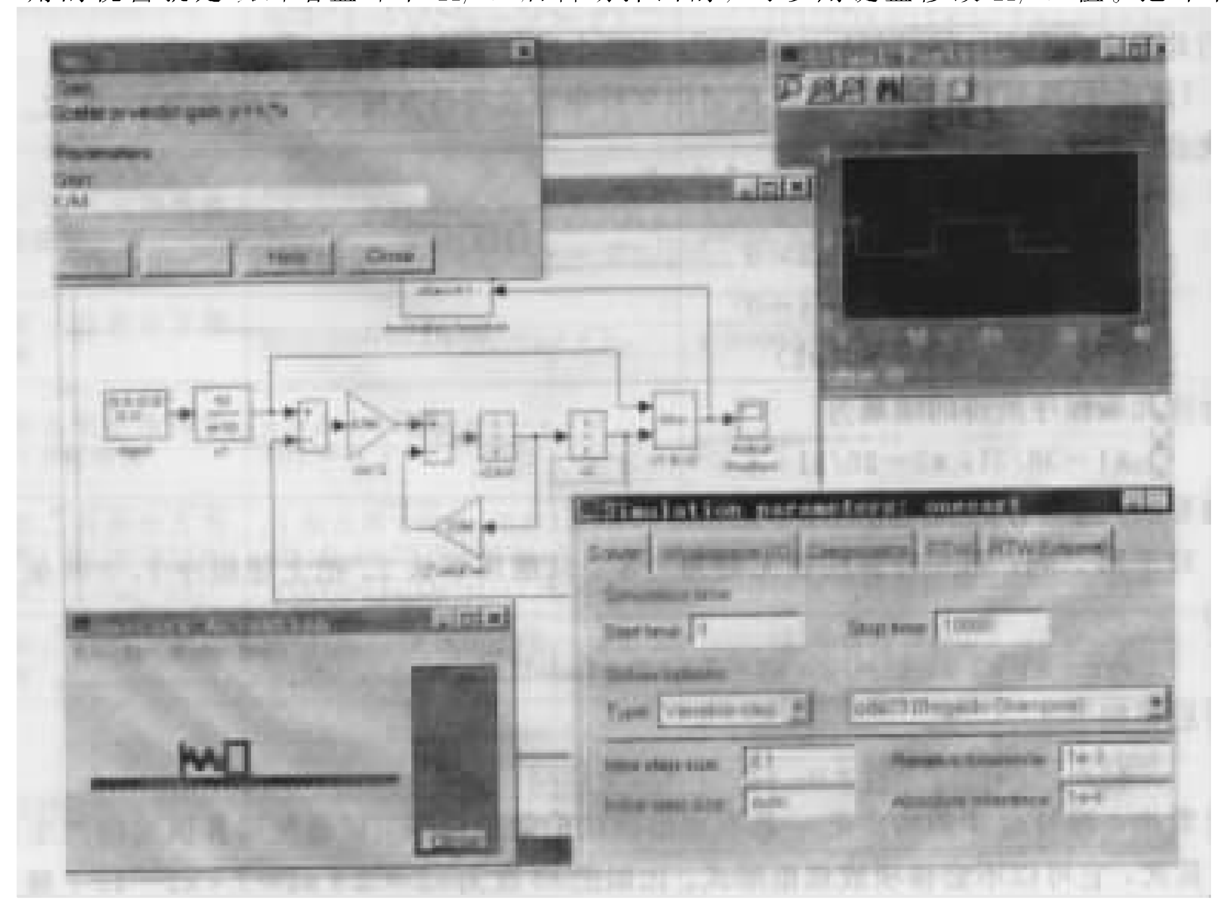


图 10－1　用 Simulink 仿真弹簧－质量系统时的部分图形界面

数设置完成后，就要设置系统的仿真参数，这时可选击主菜单上的 simulation，再选择其下拉菜单上的 parameter，屏幕上将弹出右下角的仿真参数设置窗口，设定这些参数后，就可以进行仿真了。

在 simulation 的下拉菜单上选择 start 项，就开始仿真了，从右上角的示波器上可以观察波形和数据，在左下角可以观察实体运动的动画。

Simulink 工具箱不提供新的函数，因为它全部采用图形界面，所以不会增加用户的负担，但是它会自动生成与仿真图形界面相对应的特殊文件，称为 S－文件，在初级应用中，可以不管它。

10.3 MATLAB 中专用工具箱简介

在 MATLAB toolbox 子目录下，除去已经介绍过的 MATLAB、Symbolic 和 Simulink 等三个通用工具箱外，还有数十个专用工具箱，而且数量不能增加，表 10－2 给出了其中一部份，其中大部分只要以 MATLAB 为基础就能运行，所有“模块库”则要求有 Simulink 通用工具箱的支持。在大学本科教学中，可能用到的是信号处理和控制系统两个专用工具箱，在学生版 MATLAB 中，Mathworks 公司也提供了这两个工具箱中的部分内容。国内出版的本科的控制系统和信号处理教科书中，也都涉及并介绍了这两个工具箱，至于其它的工具箱，读者在需要时，可以查阅 MATLAB 的说明书。国内近年来也出版了不少这方面的书籍可供参阅。

表 10－2 MATLAB 中的专用工具箱

序　号	工具箱名	工具箱内容
1	Simulink	系统模块仿真
2	Stateflow	状态流仿真
3	rtw	实时工作室
4	optim	最优化工具箱
5	cdma	CDMA 参考模块库
6	commblks	通信模块库
7	comm	通信工具箱
8	control	控制系统工具箱
9	dspblks	数字信号处理模块库
10	daq　daq	数据采集工具箱
11	database	数据库工具箱
12	datafeed	数据输入工具箱
13	tiddk	TI 信号处理器开发工具
14	dials	表盘和传感器模块库

续表

序　号	工具箱名	工具箱内容
15	symbolic	符号数学工具箱
16	filterdesign	滤波器设计工具箱
17	ftseries	金融时间序列工具箱
18	finance	财务工具箱
19	fixpoint	定点计算模块库
20	fuzzy	模糊逻辑工具箱
21	images	图像处理工具箱
22	Instrument	仪表控制工具箱
23	Imictrl	LMI 控制工具箱
24	compiler	MATLAB 编译器
25	map	地图工具箱
26	mpc	模型预测控制工具箱
27	motdsp	Motorola 信号处理器开发工具
28	mutools	μ 分析综合工具箱
29	nnet	神经元网络工具箱
30	ncd	非线性控制模块库
31	pde	偏微分方程工具箱
32	powersys	电力系统模块库
33	robust	鲁棒控制工具箱
34	signal	信号处理工具箱
35	splines	样条函数工具箱
36	stats	统计处理工具箱
37	ident	系统辨识工具箱
38	vr	虚拟现实工具箱
39	wavelet	小波变换工具箱

附录 A MATLAB 基本部分的函数索引

cumsum(a)
cumtrapz(a)
cylinder(u)
date(w)
datenum(w)
datestr(w)
datevec(w)
datetick(w)
dbclear(f)
dbcont(f)
dbdown(f)
dblmex(f)
dbquit(f)
dbstack(f)
dbstatus(f)
dbstep(f)
dbstop(f)
dbtype(f)
dbup(f)
ddeadv(g)
ddeexec(g)
ddeinit(g)
ddepoke(g)
ddereq(g)
ddeterm(g)
ddeunadv(g)
deal(b)
deblank(v)
dec2bin(v)
dec2hex(v)
deconv(a，r)
dec2base(v)
delaunay(r)
del2(a)
delete(f)
demo(f)
det(m)
diag(d)
dialog(x)
diary(f)
diff(a)
dir(f)
disp(k)
dlmread(j)
dlmwrite(j)
dmperm(s)
dos(f)
double(b)
dragrect(x)
drawnow(h)
dsearch(r)
echo(k)
edit(f)
editpath(f)
eig(m)
eigs(m)
ellipj(t)
ellipk(t)
ellipke(t)
else(k)
elseif(k)
end(k)
eomday(w)
eps(d)
erf(t)
erfc(t)
erfcx(t)
erfinv(t)
error(k)
errorbar(u)
errordlg(x)
errortrap(k)
etime(w)
etree(s)
etreeplot(s)
eval(k)
evalin(k)
exist(k)
exp(c)
expin(t)t
expm(m)
expm1(m)
expm2(m)
expm3(m)
eye(d)
ezplot(e)
factor(t)
fclose()
feather(u)
feof(j)
ferror(j)
feval(k)
fft(a)
fft2
fftn(a)
fftshift(a)
fgetl(j)
fgets(j)
figure(h)
filepart(j)
filesep(j)
fill(u)
fill3(q)
filter(a)
filter2(a)
find(n)
findobj(h)
findstr(v)
finite(n)
fix(c)
flag(q)
flipdim(d)
fliplr(d)
flipud(d)
floor(c)
flops(d)
fmin(e)
fmins(e)
format(f)
fopen(j)
for(k)
fplot(e)
fprintf(j)
frame2im(u)
fread(j)
frespace(d)
frewind(j)
fscanf(j)
fseek(j)
ftell(j)
full(s)
fullfile(j)
function(k)
funm(m)
fwrite(j)
fzero(e)
gallery(d)
gamma(t)
gammainc(t)(t)
gammaln(t)
gca(h)
gcd(t)
gcf(h)
gco(h)
gcbo(h)
gcbf(h)
get(h)
getenv(f)
getfield(b)
getframe(u)
ginput()
global
gmres(s)
gplot(s)
gradient(a)
gray(q)

ode113(e)
ode15s(e)
ode23(e)
ode23s(e)
ode45(e)
ones(d)
orient(p)
orth(m)
otherwise(k)
pack(f)
pagedlg(x)
pareto(u)
pascal(d)
patch(h)
path(f)
patialpath(j)
pathsep(j)
pause(k)
pcg(s)
pcode(f)
pcolor(u)
permute(b)
pi(d)
pie(u)
pie3(u)
pink(q)
pinv(m)
planerot(m)
plot(p)
plotmatrix(u)
plotyy(p)
plot3(q)
pol2cart(t)
polar(p)
poly(r)
polyarea(r)
polyder(r)
polyeig(m)
polyfit(r)

polyval(r)
polyvalm(r)
pow2(c)
ppval(r)
primes(t)
print(p)
printdlg(x)
printopt(p)
prism(q)
prod(a)
profile(f)
propedit(x)
pwd(f)
qmr(s)
qr(m)
qrdelete(m)
qrinsert(m)
quad(e)
quad8(e)
questdlg(x)
quit(f)
quiver(u)
quiver3(u)
qz(m)
rand(d)
randn(d)
randperm(s)
rank(m)
rat(t)
rats(t)
rbbox(x)
rcond(m)
readme(f)
real(c)
realmax(d)
realmin(d)
rectint(r)
refresh(h)
rem(c)

repmat(d)
reset(h)
reshape(d)
resi2(r)
residue(r)
return(k)
rgb2hsv(q)
rgbplot(q)
ribbon(u)
rjr(s)
rmfield(b)
rmpath(f)
roots(r)
rosser(d)
rot90(d)
rotate(h)
rotate3d(q)
round(a)
rref(m)
rsf2csf(m)
run(k)
save(f)
schur
script(k)
sec(c)
sech(c)
semilogx(p)
semilogy(p)
set(h)
setdiff(n)
setfield(b)
setstr(v)
setxo(n)r
shading(q)
shiftdim(b)
shg(h)
sign(c)
sin(c)
sinh(c)

size(d)
slice(u)
sort(a)
sortrows(a)
sound(a)
soundsc(a)
spalloc(s)
sparse(s)
spaugment(s)
spconvert(s)
spdiags(s)
specular(q)
speye(s)
spfun(s)
sph2cart(t)
sphere(u)
spinmap(q)
spline(r)
spones(s)
spparms(s)
sprand(s)
sprandn(s)
sprandsym(s)
sprank(s)
spring(q)
sprintf(j)
spy(s)
sqrt(c)
sqrtm(m)
squeeze(b)
ss2tf(r)
ss2zp(r)
sscanf(j)
stairs(u)
startup(f)
std(a)
stem(u)
stem3(u)
str2mat(v)

此表中各函数名后括号内的字母为库的序号。查某个函数的功能时，可有两种方法：

(1) 在本书内粗查：先根据该函数后括号内的小写字母，查表 1-2，找到相应的库名，再根据该库函数表所在的章节、页数，查找该节文字或库函数表内的说明；

(2) 在计算机上细查：键入 help 函数名，看其英文说明。

附录 B　应用实例索引

例　号	内　　容	用到的主要函数
5－1－1	单变量函数的计算和绘图	子程序 function 建立与调用，diff
5－1－2	单变量函数的极坐标绘图	polar，subplot
5－1－3	参数方程，摆线的绘制	axis
5－1－4	曲线族的绘制	hold
5－1－5	极限的定义和判别	break，while，if
5－2－1	二次曲面参数对形状的影响	NaN，surf，zlabel，all
5－2－2	空间曲面的交线	input，eval，colormap，meshgrid
5－2－3	用平行截面分析空间曲面	plot3，mesh，eps，axis，colormap
5－3－1	数列的表示方法	prod（1：n）
5－3－2	用幂级数计算指数函数	sprintf
5－3－3	多项式的泰勒级数展开	polyval，polyder，prod
5－3－4	任意函数的泰勒级数展开	diff，round
5－4－1	求任意非线性方程的解	fplot，fzero，function
5－4－2	求曲线所围面积	trapz，quad，stairs，sum
5－4－3	二重积分	trapz，fill
5－4－4	三重积分	plot3，mesh，rotate3d，meshgrid
5－4－5	求微分方程的数值解	ode23，function，legend
5－5－1	矩阵的初等变换，高斯消元法	dct，rank，tracc
5－5－2	矩阵的条件数与解的精度	hilb，cond
5－5－3	对称矩阵的特征方程和特征根	cig，dct，tracc，cyc
6－1－1	简单单位换算	num2str
6－1－2	多种单位换算	fprintf，input
6－1－3	数据拟合	polyfit
6－2－1	真空中的抛射体运动	roots
6－2－2	x，y 两方向运动合成和求动量矩	input（：，'s'），diff
6－2－3	斜面物体下滑	解线性方程组

续表一

例　号	内　　容	用到的主要函数
6－2－4	碰撞问题	roots
6－3－1	麦克斯韦气体分子速度分布率	fill，trapz，function
6－3－2	热力学过程及 P—V 图	switch－case，while，menu，sprintf
6－4－1	平面上 N 个电荷的库仑引力	fprintf，for
6－4－2	线电荷的电位计算	sum
6－4－3	电位分布及电场计算	gradient，eval，contour，quiver，meshc
6－5－1	山毕奥—萨伐定律计算磁场	quiver，sum
6－5－2	亥姆霍兹线圈的磁场验算	mesh，format ＋
6－6－1	振动合成和拍频现象	sound
6－6－2	多普勒效应的演示	sound
6－7－1	双缝光干涉	image，colormap，gray
6－7－2	光的单缝衍射	sum，用数组列方程
7－1－1	平面力系的简化	向量用数组计算
7－1－2	求平衡杆系的支撑反力	解线性方程组
7－1－3	导弹追踪运动目标的轨迹	数值微分和积分，ode23，diff
7－1－4	四连杆运动的分析计算	fzero，复杂方程求解
7－1－5	考虑空气阻力时的抛射体运动	四阶的 ode45，find
7－1－6	圆柱体平面运动的分析	解联立方程
7－2－1	静不定杆系的受力计算	解线性方程组
7－2－2	悬臂梁的变形计算	cumsum，不连续函数表达法
7－2－3	双铰支梁的变形计算	cumtrapz，求积分常数
7－2－4	复杂应力状态的分析 Moore 圆	复杂方程作图
7－2－5	拉弯合成的强度设计	roots
7－3－1	单自由度振动阻尼系数的影响	复合图形，mesh
7－3－2	单自山度阻尼系统的强迫振动	conv，residue
7－3－3	二自由度可解耦系统无阻尼振动	eig，特征根分解
8－1－1	直流电路的稳态计算	解线性方程组
8－1－2	直流电路的暂态计算	图柄参数设置，set
8－1－3	交流电路的稳态计算	复数线性方程，compass（MATLAB 4），set
8－1－4	交流电路加多频信号	元素群运算，复数及相量运算
8－1－5	交流电路的稳态计算	复数线性方程，compass（MATLAB 5）

续表二

例　号	内　　容	用到的主要函数
8－1－6	谐振回路计算	loglog，semilogx，find
8－1－7	网络频率特性的计算	subplot，元素群计算
8－1－8	网络参数的计算与变换	inv，det
8－2－1	二极管特性曲线的绘制	line，fill，ginput，画线路图，legend
8－2－2	放大器低频区频率特性	用 for 语句分析参数影响
8－2－3	运算放大器有限增益对闭环频率响应的影响	poly，polyval，semilogx
8－3－1	可控硅点火角与电压的关系	不连续函数表达法，sum，legend，希腊字标法
8－3－2	三相感应电机旋转磁场的形成	drawnow，动画程序编法
8－3－3	三相感应电机转矩特性曲线	plot 语句中设定线宽
8－4－1	同轴线特性阻抗计算	log，legend
8－4－2	Smith 阻抗图	用复数 plot 语句画图
8－4－3	调幅、调频、调相波形及频谱	fft，axis
9－1－1	连续信号的 MATLAB 表示法	stairs
9－1－2	系统的冲击函数和卷积	residue
9－1－3	阻尼系数对过渡函数的影响	residue，poly
9－1－4	求线性系统零输入响应	vander，rot90，roots
9－1－5	多级放大器的上升时间	prod，unwrap，logspace，log10 重根处理方法
9－1－6	用傅里叶级数合成方波	多条曲线叠合，mesh
9－1－7	方波经过低通滤波器后恢复	傅里叶变换和反变换算式，fliplr
9－2－1	离散信号的表示方法	stem，real，imag，逻辑式表示信号
9－2－2	差分方程的递推求解	int2str，沿时间轴平移变量
9－2－3	用矩阵乘法表示离散傅里叶变换	fft，nextpow2，randn
9－3－1	传递函数和零极点的关系	poly，roots，residue
9－3－2	动态环节的串联、并联和反馈	conv，自定义 polyadd 函数
9－3－3	齐次状态方程解状态转移矩阵	三维矩阵 F（：，：，：），reshape，squeeze
9－3－4	零极点分布对系统特性的影响	三维矩阵 P（：，：，：）
9－4－1	偏微分方程的数值积分	迭代方法

参 考 文 献

1 The Mathworks Inc.. The Student Edition of MATLAB Version 4 User's Guide. Prentice-Hall, Inc., 1996

2 The Mathworks Inc.. Using MATLAB Version 5. 1997.1

3 The Mathworks Inc.. Using MATLAB Graphics Version 5. 1996.12

4 楼顺天，于卫，阎华梁. MATLAB程序设计语言. 西安：西安电子科技大学出版社，1997

5 Thomas L. Harman, James B. Dabney & Norman John Richert. Advanced Engineering Mathematics Using MATLAB. PWS Publishing Company, 1997

6 Alejandro L. Garcia & Cecile Penland. MATLAB Projects for Scientists and Engineers. Prentice-Hall, 1996

7 Louis H. Turcotte & Howard B. Wilson. Computer Applications in Mechanics of Materials Using MATLAB. Prentice-Hall, 1998

8 Ong, hee-Mun. Dynamic Simulation of Electric Machinery Using MATLAB/Simulink. Prentice-Hall, 1998

9 William T. Thomson & Marie Dillon Dahleh. Theory of Vibration with Applications, 5e. Prentice-Hall, 1998

10 John R. Buck, Michael M. Daniel & Andrew C. Singer. Computer Explorations in Signals and Systems Using MATLAB. Prentice-Hall, 1997

11 陈怀琛，黄道君. 控制系统CAD及MATLAB语言. 北京：电子工业出版社，1996

12 Vinay K. Ingle, John G. Proakis. 数字信号处理及其MATLAB实现. 陈怀琛，王朝英，高西全译. 北京：电子工业出版社，1998

13 薛定宇. 控制系统计算机辅助设计——MATLAB语言及应用. 北京：清华大学出版社，1996

14 张志涌，刘瑞桢，杨祖樱. 掌握和精通MATLAB. 北京：北京航空航天大学出版社，1997

15 施阳，李俊等. MATLAB语言工具箱——TOOLBOX实用指南. 西安：西北工业大学出版社，1998

16 楼顺天等. 基于MATLAB的系统分析与设计. 西安：西安电子科技大学出版社，1998

17 Duane C. Hanselman, Bruce Littlefield. 精通MATLAB——综合辅导与指南. 李人厚，张永安译. 西安：西安交通大学出版社，1998

18 张培强主编. MATLAB语言——演算纸式的科学工程计算语言. 合肥：中国科学技术大学出版社，1995

19 Mount Holyoke College. Laboratories in Mathematical Experimentation－A Bridge to Higher Math, New York: Springer, 1977

20 Phillip Kent, et. al. Experiments in Undergraduate Mathematics－A Mathematica Based Approach, Imperial Colloge Press, London, 1996

21 Landau, Rubin H. Computational Physics－Probleming Solving with Computers, New York: John Wiely & Sons, Inc., 1997

22 John O. Attia. Electronics and Circuit Analysis Using MATLAB. CRC Press, 1999

23 陈怀琛，吴大正，高西全. MATLAB在电子信息课程中的应用(第二版). 北京：电子工业出版社，2003

24 陈怀琛. 数字信号处理教程——MATLAB释义与实现. 北京：电子工业出版社，2004